TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

ROPE SIDEBRAS
TESTAUSPROSESSIN KEHITTÄMINEN OHJELMISTON KÄYTTÖÖNOTTOPROJEKTISSA

Diplomityö

Tarkastaja: professori Hannu Jaakkola
Tarkastaja ja aihe hyväksytty
Tuotantotalouden ja rakentamisen
tiedekunnan tiedekuntaneuvoston
kokouksessa 5. kesäkuuta 2013

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Tietotekniikan koulutusohjelma
SIDEBRAS, ROPE: Testausprosessin kehittäminen ohjelmiston käyttöönottoprojektissa
Diplomityö, 80 sivua, 39 liitesivua
Elokuu 2013
Pääaine: Ohjelmistotekniikka
Tarkastaja: professori Hannu Jaakkola
Avainsanat: Altéa, dokumentointi, testaus, käyttöönotto, projekti, testitapauksien suunnittelu ja toteutus, testausprosessin kehittäminen ja hallinta, TMMi

Ohjelmistotuotteen käyttöönottoon liittyy monta eri vaihetta. Erityisesti käyttöönoton suunnitteluvaiheeseen on panostettava, jotta käyttöönoton aikana osataan varautua mahdollisiin riskien aiheuttamien häiriöiden minimointiin. Avainasemassa käyttöönotossa on itse ohjelmisto, joka tiettyjen välietappien avulla viedään tuotantoon. Välietappien avulla voidaan varmistaa, että kaikki kriittiset ja vähemmän kriittiset seikat on otettu ohjelmiston käyttöönoton osalta huomioon. Ohjelmiston suunniteltu ja laadukas testaaminen ja testauksen johtaminen takaavat sen, että ohjelmisto on käyttöönotettavissa. Ohjelmistoprojektin käyttöönottovaiheessa tulee huomioida asennettavan ohjelmiston käyttötarkoitus ja toiminnallisuudet, käyttöönottoympäristö, käyttöönoton laajuus, ohjelmistointegraatiot, käyttöönoton tuki sekä käyttöönottoprojektin aikataulutus. Tärkeänä osana projektia on myös projektiin osallistuva sitoutunut henkilöstö, jonka ammattitaidon avulla projektien läpivienti on mahdollista.

Käyttöönottoprojektissa käytettiin useita erilaisia testausmetodeita onnistuneen lopputuloksen saavuttamiseksi ja ennen kaikkea virheiden poissulkemiseksi ohjelmistosta. Testauksen avulla oli myös mahdollista testata ohjelmiston ohella lentokoneen kääntöön liittyvien prosessien toimivuutta. Yhdistimme useita testitapauksia toisiinsa, jolloin voitiin simuloida kaikki lennon kääntöön liittyvät vaiheet. Testauksen laajuus ei ollut projektin alkaessa täysin selvä. Projektissa ei myöskään vältytty muutoshallinnalta ja ohjelmistospesifikaation puutteilta, minkä takia toiminnallisuuksien toteutus viivästytti käyttöönoton aikataulua.

Työn tavoitteena oli selvittää, miten yrityksessä, jossa ohjelmiston testaus ei ole strategisesti merkittävässä roolissa liiketoiminnan kannalta, voidaan kehittää testausta sen hallittavuutta, testausprosessia sekä testausprosessin hallintaa tulevia vastaavia ohjelmistoprojekteja varten. Osoitan tässä työssä projektissa saavutetun testausprosessin kypsyystason ja arvioin mitkä kokonaisuudet testauksen prosessin kehittämisessä saavutimme. Projektissa testauksen prosessi kehittyi lähes olemattomista lähtökohdista Test Maturity Model integration -mallin (TMMi) viisiportaisen mallin tasolle kolme (3): toimintatavat. Arviointikriteereissä käytin TMMi-mallia hyväksi TMMi Assessment Method Application Requirements -ohjeistuksen (TAMAR) avulla.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
SIDEBRAS, ROPE: Testing Process Development in a Software Deployment Project
Master of Science Thesis, 80 pages, 39 Appendix pages
August 2013
Major: Software Engineering
Examiner: Professor Hannu Jaakkola
Keywords: Altéa, Documentation, Testing, Deployment, Project, Test Case Planning and Execution, Test Process Development and Management, TMMi

The software deployment project consists of many phases. The planning phase of a deployment needs special attention so that possible risks and other interruptions can be minimised. The software product has a key role in the deployment project. The software product can be utilised in the production environment after certain stages have been completed. By mapping out the steps of a deployment project it is possible to ensure that all the major and less critical details have been covered. Software design, the quality of the testing procedures and proper management of the testing phase will ensure that the software product can be deployed. There are several issues to consider related to the deployment phase of the project: software feasibility and functionalities, the deployment environment and scope of the deployment, software system integrations, deployment support and the planned schedule of the deployment project. Committed and competent staff is important to the success of the project.

In the deployment project several testing methods were used to achieve a workable final result and to omit possible errors from the end product. With the help of testing procedures, it was possible to test the functionalities of both the software and aircraft turn around processes. We combined several different test cases together and executed them in real time. This way it was possible to simulate all the phases and tasks involved in aircraft turn around process. When the project started, the scope of the testing was not completely clear. Change management was also needed during the project. Some of the software functionalities were not produced as they were designed, so those functionalities had to be reproduced and this delayed our deployment schedule.

The aim of this thesis was to research and prove how a non-software company, where software testing does not hold a strategically significant role for its business, can improve testing procedures as well as management of testing and test processes management in order to utilise the findings in corresponding projects. In this thesis I demonstrate the testing level that was achieved within the project and evaluate what tasks in the test process development were achieved. During the project the testing process developed from level zero to Test Maturity Model integration (TMMi) level three (3): Defined, on a scale of one to five. I used the TMMi model together with the TMMi Assessment Method Application Requirements (TAMAR) instructions when assessing the testing process.

ALKUSANAT

Tein diplomityön varsinaisen työosuuden työskennellessäni Finnairilla järjestelmäasiantuntijana vuosina 2007–2009. Varsinainen testauksen suunnittelu, dokumentointi ja itse testaus ja sen hallinnointi toteutettiin vuoden 2007 maalis- ja marraskuun välillä. Työn tarkastajana toimi Hannu Jaakkola Tampereen teknillisestä yliopistosta, jota haluan kiittää kärsivällisyydestä työn valmistumisen osalta. Diplomityön ohjaaja on Finnairin puolelta toiminut esimieheni ja projektin johtaja Kari Pauro. Olen myös keskustellut testausprosessista intialaisten kollegoideni sekä useiden muiden tahojen kanssa Finnairilla, kuin myös nykyisissä työtehtävissä Innofactorilla. Nykyinen toimenkuva on antanut myös uutta näkökantaa työn teoriaosuuteen.

Idean ja aiheen työhön sain siitä ajatuksesta, että vaikka Finnair ei ole ohjelmistotalo tai ohjelmistokehityksen osa-alueella toimiva yritys, tulee myös Finnairilla olla selkeä kuva testauksesta ja testausprosessin luonteesta. Testaussuunnitelmien hyväksyminen, testauksen dokumentointi, aikataulutus, yhteydenpito eri alihankkijoiden kesken ja alihankkijoiden ohjeistaminen ja johtaminen jäävät kuitenkin suurimmaksi osaksi useassa projektissa Finnairin vastuulle. Testausprosessista ja testauksesta on hyvä olla yrityksessä kattava malli ja ohjeistus kuvaamaan, että mitä kaikkea asian osalta tulee ottaa huomioon vastaavissa ohjelmistoprojekteissa. Prosessin ja ohjeistuksen avulla Finnairin edustajilla on mahdollisuus hallita testaukseen liittyviä kokonaisuuksia ja edelleen kehittää testausprosessiin liittyvää dokumentaatiota tarvittaessa.

Varsinainen Altéa FM -projekti, jossa työskentelin yhtenä järjestelmäasiantuntijana, liittyi ohjelmistosovellukseen, jolla toteutetaan lentojen tasapainolaskelmia useita eri tekijöitä huomioiden. Ohjelmisto kerää ja siihen syötetään dataa maailmanlaajuisesti useista lähteistä. Näistä lähteistä osa on toisia ohjelmistoja ja osan datasta järjestelmään syöttää ohjelmiston käyttäjä. Suurin haaste hankkeessa oli muiden ohjelmistojen integrointi ja ohjelmistoista siirtyvien tietueiden siirron varmistus ja erityisesti oikeanlaisen tiedon siirto Altéa FM -sovellukseen. Ohjelmistojen integrointi on nykyisin ohjelmistojen osa-alueella tärkeää tiedon siirtyessä lähdejärjestelmästä ja tiedon tuottajalta suoraan käytettävään järjestelmään. Finnair käytti myös osalle ohjelmiston testausta alihankintaa. ITC Infotechin testaajat suorittivat suurimmaksi osaksi ohjelmiston perustestauksen ohjelmiston määritysdokumenttien avulla muodostetuilla testitapauksilla. He jatkavat edelleen testauksia meidän suunnittelemiemme testitapausten pohjalta uuden ohjelmistoversion ilmestyessä. Tällöin suoritetaan regressiotestaus ja tarvittaessa ohjelmistoon kohdistuvien muutosten testaaminen ennen uuden version käyttöönottoa tuotantoon.

Kiitokset haluan esittää erityisesti Kari Paurolle. Lisäksi haluan kiittää Veikko Vuorelaa, joka toimi Altéa FM -sovellus projektissa lähimpänä esimiehenä. Erityiskiitos kuuluu myös Tero Lehtorannalla ja Tuija Makkoselle. Kiitokset kuuluvat myös muille testaus- ja projektitiimille ja kaikille tahoille, joiden kanssa olen saanut olla tekemisissä projektin aikana ja sen jälkeen. Olette antaneet paljon, toivottavasti me kaikki olemme myös oppineet paljon toisiltamme.

23.6.2013 Helsingissä

Rope Sidebras
rope@iki.fi

SISÄLLYS

TERMIT JA NIIDEN MÄÄRITELMÄT

Altéa CM	Altéa Customer Management (Altéa CM) -ohjelmisto lennon lähtöselvitystä ja porttipalveluita varten, jossa on liitokset Altéa Inventory -ohjelmistoon ja varausjärjestelmään.
Altéa FM	Altéa Flight Management (Altéa FM) -ohjelmisto lennon valmistelua ja tasapainolaskelmien toteuttamista varten. Amadeuksen tuote.
Altéa Inventory	Altéa Inventoryn kautta lentojen perustiedot ja myyntitiedot asetetaan varausjärjestelmään ja edelleen siirtyvät Altéa FM/Altéa CM -ohjelmistoihin.
Altéa-ohjelmistoperhe	Altéa-ohjelmistojen ohjelmiston muodostuvat Altéa Admin -ohjelmistosta, Altéa Inventory -ohjelmistosta, Amadeus Reservations -ohjelmistosta, Altéa Flight Management -ohjelmistosta ja Altéa Customer Management -ohjelmistoista.
Amadeus Reservations	Järjestelmä, jonka kautta lentojen matkustajien varaukset lennoille tehdään ja niitä voi hallinnoida tarvittaessa.
Amadeus	Altéa-ohjelmiston toimittajataho, joka on ilmailualalla ja muulla logistiikan alueella merkittävä toimija maailmanlaajuisesti. Pääkonttori sijaitsee Espanjassa, Madridissa ja Amadeus työllistää n. 10 000 henkilöä maailmanlaajuisesti.
ARINC	ARINC on lentokenttien suljettujen verkkoympäristöjen omistajataho. Verkkoympäristö on nimeltään ARINC Data Network Service. ARINC tunnetaan myös ilmailualan verkkoteknologiaan erikoistuneena yrityksenä. Tässä työssä ARINC:lla tarkoitetaan lentokentän suljettua verkkoympäristöä.
Black-box -testaus	Mustalaatikkotestaus. Testaajalla on käytettävissään ohjelmiston spesifikaatiot, ohjelmiston arkkitehtuurikuvaukset tai ohjelmistoon liittyvää dokumentaatio. Testaaja testaa ohjelmistoa mustan laatikon

lähestymistavan mukaisesti, eli tuntematta ohjelmiston toteutusta.

CMMI Capability Maturity Model Integration, eli prosessien kehittämiseen toteutettu kyvykkyys- ja kypsyysmalli, joka ohjaa prosessien kehittämistä erilaisilla prosessialueilla organisaatiossa. Yleensä mallin hyödyntämiseen liittyy askelmien eli portaiden saavuttamiseen. CMMI on kehitetty Carnegie Mellon yliopistossa.

End-to-end testaus Järjestelmän testaaminen, jossa testataan ohjelmiston käyttö kaikkien erilaisten perustoiminnallisuuksien osalta yhdessä tai useammassa testisessiossa. Testaus kestää niin kauan kuin se luonnollisesti kestäisi myös todellisuudessa.

FIAs eli Finnairin integraatiosovellukset

Finnair Impacted Applications, eli FIAs, joista tiedot siirretään ohjelmistointegraation avulla tai viestiliikenteen avulla Altéa-järjestelmään. Nämä ohjelmistot ovat rahdin eGO-ohjelmisto, Finnairin lähtöselvitysjärjestelmä: AY FIDO, lennon reitityksen ja lentopetrolin määrän laskeva ohjelmisto: LIDO, sekä lentopetrolin tilausjärjestelmä kentällä: FUEL CIS. Lentokoneiden rotaatiota ja suunnittelua ohjaava ohjelmisto on nimeltään SCOPE ja ohjaamon viestiliikennettä hallinnoin oma ohjelmisto: HERMES.

Finnair Suomen kansallinen lentoyhtiö.

Gray box -testaus Harmaalaatikkotestaus. Testaajalla on käsitys ohjelmiston toimintalogiikasta ja testitapauksia voidaan suunnitella ja toteuttaa näiden perusteella tarkemmin.

Hyväksymistesti Asiakkaan suorittama testi, jossa järjestelmää verrataan vaatimusmäärittelyyn.

IEEE-829 Institute of Electrical and Electronics Engineers -organisaation (IEEE) julkaisema standardi ohjelmistotestaukseen. Vaatimuksina tässä standardissa on muun muassa testaussuunnitelma, testitapausten suunnittelu

	ja toteutus testidatalla, raportointi ja poikkeamat testauksessa ja loppuraportti.
IEEE-1012	Institute of Electrical and Electronics Engineers (IEEE) toteuttama standardi ohjelmistotestaukseen ja erityisesti verifioinnin ja validoinnin suorittamiseen ja ohjaamiseen.
IEEE-standardit	Institute of Electrical and Electronics Engineers (IEEE) laatii ohjeistuksia ja parhaisiin käytäntöihin liittyviä määrityksiä eli standardeja, joiden noudattaminen on todettu hyväksi tavaksi toimia.
Integrointitesti	Järjestelmän tai sen osien testaaminen kokonaisuutena sen varmistamiseksi, että osat toimivat yhdessä täydellisenä kokonaisuutena.
Integrointiympäristö	Erillinen testausympäristö, jossa ohjelmistointegroinnit ja integraatiot pyritään testaamaan ennen kuin ne siirretään varsinaiselle testille ja tuotannon beta-versioon.
ISO-15504	International Organization for Standardization (ISO) kansainvälisten standardien julkaisija. ISO-15504 on prosessien arviointistandardi, jonka avulla voidaan arvioida eri toiminnan osa-alueita. ISO-15504-2 on erityisesti keskittynyt ohjelmisto ja IT-toimintojen arviointiin.
Java	Nykyaikainen Oraclen kehittämä ohjelmointikieli, jolla on mahdollista toteuttaa itsenäisiä ohjelmistoja ja ohjelmiston toiminnallisuuksia myös hajautettuun verkkoon ja www-selaimille.
Järjestelmätesti	Järjestelmän testaaminen ennen sen toimittamista asiakkaalle sen varmistamiseksi, että järjestelmä on vaatimusmäärittelyn ja sopimuksen mukainen.
Parallel Run	Parallel run eli rinnakkaistestaus. Ohjelmistoa ajetaan rinnan nykyisen järjestelmän kanssa tosipohjaisella datalla ja reaaliaikaisesti lopullisessa tuotantoympäristössä, jolloin sen käyttö todetaan rinnan olemassa olevan ohjelmiston kanssa.

Prosessi Jatkuva toimintamalli, jossa on joukko toisiinsa liittyviä toimintoja ja joka on vakiintunut organisaatioon ja jota voidaan kehittää. Prosessilla on omistaja ja sen suorituksena tulee aina jokin lopputulos.

PRD, tuotanto-ohjelmisto Varsinainen tuotantosovellus, jota käytetään tuotannossa, eli lopullinen käytössä oleva ohjelmisto. PRD = production, eli tuotanto.

Qantas Australian kansallinen lentoyhtiö.

RESA RESA on lentokenttien suljettujen verkkoympäristöjen omistajataho ja verkkoympäristö. RESA tunnetaan myös ilmailualan verkko- ja ohjelmistoteknologiaan erikoistuneena yrityksenä. Tässä työssä RESA:lla tarkoitetaan lentokentän suljettua verkkoympäristöä.

Self Service Check in Kiosk

Lähtöselvitysjärjestelmä, joka on omatoimisesti käytettävissä lentokentillä erillisissä kioskisovelluksissa eli systeemi sisältää sekä ohjelmiston että laitteiston.

SITA SITA on lentokenttien suljettujen verkkoympäristöjen toimittaja ja SITA-verkkoympäristö. SITA tunnetaan myös ilmailualan verkko- ja ohjelmistoteknologiaan erikoistuneena yrityksenä. Tässä työssä SITA:lla tarkoitetaan lentokentän suljettua verkkoympäristöä.

SITA-sanoma TELEX-sanoma, jota lentoteollisuudessa kutsutaan nimellä SITA-sanoma. SITA-sanomat voivat olla määrämuotoisia tietyn formaatin mukaan tuotettuja tai sitten vapaamuotoisia. Tunnetaan myös SITATEX-formaatissa.

Scrum Ketterä ohjelmistokehitysmenetelmä, jossa ohjelmisto tai sen kokonaisuudet toteutetaan pyrähdyksissä, eli sprinteissä ja kehitykseen valitaan ennen kunkin sprintin alussa kohteet, jotka kehitetään loppuun kunkin sprintin aikana.

Systeemi-integraatiotestaus

Useamman ohjelmiston välisen liittymän eli integraatioiden tai integrointien testaaminen. System Integration Testing (SIT).

TAMAR TMMi Assessment Method Application Requirements (TAMAR) määrittelee vaatimukset arviointia varten TMMi:n kypsyystasojen suhteen.

Testaus Ohjelman (tai sen osan) suorittaminen tarkoituksena löytää virheitä. Usein ohjelmaa pyritään käyttämään väärin määritykseen tai ohjeistukseen nähden.

Testausstrategia Korkean tason dokumentti, joka sisältää ylätasolla testauksen lähtökohdat, tavoitteet laajuuden, testausalueet, testauksen hallinnan ja johtamisen, dokumentaation laajuuden ja riskit. Testausstrategia on dokumentti testauksen tavoitteen kirkastamiseksi.

Testaussuunnitelma Suunnitelma, jossa kuvataan testauksen tavoitteet, aikataulu, testausvälineet, vastuut, raportointikäytännöt, riskit, sekä rajaukset sen osalta, että mitä ei testata. Master Test Plan on pohjadokumentti, jonka avulla testaussuunnitelmia voidaan toteuttaa.

TMMi TMMi (Test Maturity Model Integration) on kypsyystasojen avulla organisaation testauksen ja testausprosessin arviointiin kehitetty malli. Sen tarkoituksena on parantaa testausprosessin kypsyyttä organisaatiossa. Mallia voidaan käyttää esimerkiksi CMMI:n rinnalla. Mallista vastaa TMMi-säätiö.

TRN TRN = Training, eli koulutusympäristö, missä koulutukset käyttäjille suoritettiin.

Tuotannon beta-versio Ennen tuotanto-ohjelmiston käyttöönottoa julkaistua ohjelmistosovellusta pidetään tuotannon beta-versiona, jonka avulle toiminnallisuudet validoidaan.

Yksikkötesti	Yksittäisten moduulien testaaminen heti kun ne ovat suorituskelpoisia; katettava mahdollisimman tarkkaan ohjelmakoodin eri osat.
UAT, testiohjelmisto	Testaukseen käytetty sovellus, jossa testaus tehdään ennen tuotannossa verifiointia. UAT = User Acceptance Testing Environment, eli järjestelmätestaukseen julkaistu ohjelmistoversio.
UML	Unified Modelling Language (UML) on kuvausmalli, jolla ohjelmiston erilaiset keskinäiset suhteet kuvataan. UML-kaavioiden avulla on mahdollista ymmärtää ohjelmiston toiminnallisuudet kuvattuna dokumentaatiossa.
Validointi	Ohjelman suoritus todellisessa ympäristössä tarkoituksena löytää kohtia, joissa ohjelma toimii käyttäjän tarpeiden vastaisesti.
Verifiointi	Ohjelman toiminnan tarkastaminen ohjelman määrityksiin nähden.
Vesiputousmalli	Ohjelmistokehitysmalli, jossa eri vaiheet seuraavat toisiaan vaihejaon periaatteella. Vesiputousmalli sisältää seuraavat työsuoritukset seuraavassa järjestyksessä: esikartoitus, määritys, suunnittelu, toteutus, integrointi ja testaus ja käyttöönotto ja ylläpito.
V-malli	Testauksen V-mallissa testaus suoritetaan alatasolta ylätasolle, eli ensin tehdään yksikkötestaus, tämän jälkeen integrointitestaus moduulien kesken ja lopuksi suoritetaan järjestelmätestaus ja edelleen mahdollisuuksien mukaan vielä hyväksymistestaus.

1 JOHDANTO

Projektissa, jossa työskentelin Finnairilla vuosina 2007–2009, tehtiin kaksi laajaa ohjelmiston käyttöönottoa. Projektin merkittävistä kokonaisuuksista yksi oli testaus. Projektin aikana toteutettiin testauksen hallintaan, ohjaamiseen ja tehostamiseen toimintatapoja, joiden avulla oli madollista kehittää myös testausprosessia. Testausprosessin kehittäminen yrityksen sisäiseen käyttöön, ohjelmistokehityksen testauksen toteuttamiseen ja alihankkijoiden ohjeistamiseksi, oli tärkeää, jotta toimintatavat saatiin yhteneviksi henkilöstön ja eri toimijoiden kesken. Toteutettuja dokumentteja ja toimintatapoja vertasin testausprosessin malliin, joka on TMMi Foundation kehittämä TMMi-malli (Test Maturity Model Integration). Tämän. mallin avulla pyrin arvioimaan projektissa testauksen prosessien kypsyyttä, joka projektin aikana saavutettiin. Yhteiset toimintatavat nopeuttivat testausta ja paransivat testauksen laatua. Vaikka vertailukohtana oli valmis testauksen prosessimalli, jonka avulla testausta ja suunnittelua voi ohjata, tehtiin Finnairilla testaukseen liittyvä kehittäminen osana projektia. Testauksen kehittäminen auttoi projektin läpiviemisessä, kun testauksen dokumentaation toteuttamiselle oli standardimallit ja dokumenttipohjat. Testausprosessiin liittyen on edelleen tarvittaessa mahdollista tehdä kehitystoimenpiteitä, jatkokehittämistä ja prosessien parantamista.

Mukana testaamassa Altéa FM -projektissa oli neljän hengen testaustiimi Finnairilta ja alihankkijoita erilaisten kokonaisuuksien testaamisessa. Itse keskityin testauksen ohella uuden oppimiseen ja projektin sekä prosessin dokumentointiin ja kehittämiseen, testauksen suunnitteluun ja testausprosessin ymmärtämiseen ja kehittämiseen. Autoin myös testauksen koordinoinnissa alihankkijan kanssa. Vastuullani oli myös testitapausten tuottaminen ohjelmiston yhden moduulin perustestaukseen ja yhden Finnairille spesifisen ohjelmiston ja varsinaisen sovelluksen yhteenliittymän, eli ohjelmistointegraation testaaminen. Liitoksien eli integraatioiden testausvastuu oli jaettu kunkin ohjelmiston osalta yhdelle testaajalle Finnairin testaustiimin neljästä jäsenestä. Systeemi-integrointien testauksen peruslähtökohtina olivat myös lähdejärjestelmän toiminnallisuuksien, niissä sijaitsevan tiedon ja sen merkityksen ymmärtäminen.

Testitapausten tuottamisessa käytettiin monia erilaisia tekniikoita. Koska käytettävissä ei ollut lähdekoodia, rakensimme perustestitapaukset ohjelmistomäärityksen avulla. Erilliset sovellusten testitapaukset kirjoitimme sen perusteella millainen kokemus kunkin sovellusalueen toiminnasta oli ja millaista tietoa sovelluksesta pitäisi testaamamme ohjelmistoon tulla. Tässä tapauksessa voidaan puhua Black-box tai Gray box-testaamisesta, sillä ohjelmiston määritykset olivat

käytettävissämme koko projektin ajan. Lähtökohtaisesti ohjelmisto testattiin Finnairin liiketoimintojen tarpeista, ja lentokoneen käännössä todellisuudessa tehtävät toiminnot piti pystyä myös uudella ohjelmistolla suorittamaan. Lopullisen hyväksymistestaus, eli end-to-end -testauksen testitapaukset toteutimme tapahtumaketjun kokonaisprosessia silmällä pitäen ja yhdistelemällä sovellusalueiden testitapauksia toisiinsa tietyn tyyppisen skenaarion mukaisesti. Tätä ennen ja tämän jälkeen Altae-ohjelmistolla tehtiin myös rinnakkaistestausta.

Luvussa 2 kerron projektista, jossa työskentelin ja pyrin kuvaamaan toimintaympäristön, kuin se on viitekehyksen näkökulmasta tarpeellista. Näin saadaan kuva siitä millaiset lähtökohdat testausprosessin kehittämiselle olivat. Luvussa on myös kuvattu toimintatapoja ja eri kokonaisuuksia, joita projektissa ja osa-projekteissa oli mukana. Altéa-projekti muodostui kahden ohjelmistotuotteen käyttöönotosta, eli Altéa Flight Management (Altéa FM) ja Altéa Customer Manager (Altéa CM). Tässä luvussa on myös selvitetty testatun ohjelmiston toiminnallisuudet korkealla tasolla ja kuvattu näiden osalta tarvittavat integraatiot muiden ohjelmistojen kanssa. Testauksessa käytetty metodologia lähti pitkälti projektin näkökulmasta ja kehittyi koko Altéa-projektin aikana mitattavaksi ja ohjattavaksi toiminnaksi. Lähtökohdat testaukselle olivat suhteellisen hyvät, sillä käytettävissä oli laaja dokumentaatio ja ohjelmiston ohjeistus, joka tosin alkuun oli melkoisen puutteellinen. Muutoshallinnan kautta tilattujen lisätöiden tekemiseltä ei kuitenkaan tässäkään projektissa voitu välttyä. Projektit sisälsivät testauksen lisäksi koulutusvalmistelut ja koulutukset, asennukset ja käyttöönoton sekä käyttöönoton tukemisen.

Luvussa 3 on kahden erilaisen projektimallin teoria lyhyesti kuvattuna, eli vesiputousmallin ja Scrum-mallin näkökulmat ohjelmistokehitystyöhön. Erityisesti näiden mallien esittelyssä on keskitytty esittelemään testauksen suorittaminen kummassakin mallissa. Projektissa toteutettiin testaus vesiputousmallin mukaisesti, sillä Altéa-projektin kokonaistarkoitus oli juuri testata valmis ohjelmisto niin, että se oli erityisesti Finnairilla käyttöönotettavissa. Esittelen myös lyhyesti erilaiset testausprosessien kehitysmallit ja standardit ja avaan tarkemmin TMMi-mallin testausprosessin kehittämisen ja mittaamisen osalta. TMMi:n jakaantuu prosessin kypsyyden mukaan viidelle eri tasolle ja projektissa kehitettyjen käytäntöjen, osaamisen ja dokumentaation määrä on peilattu mallin teorian viitekehykseen. TMMi:n mallin tasot ovat lähtötaso, hallinta, toimintatavat, mittaaminen ja kehittäminen. Testausprosessin kyvykkyyden arviointi on suoritettu pitkälti omaan kokemukseeni perustuen luvussa viisi. Arvioinnissa olen käyttänyt hyödyksi testausprosessin kypsyystason arviointia peilaamalla sitä projektin dokumentaation ja testauksen toteutukseen projektissa. Varsinainen arviointi suoritetaan TAMAR-kehikon tai -ohjeistuksen avulla (TMMi Assessment Method Application Requirements), jolloin koko arviointi dokumentoidaan ja arvion tekevät tahot ovat sertifioituja.

Luvussa 4 on testauksen teorian perusasioita ja niitä verrataan Altéa-projektissa suoritettuun testaukseen. Esittelen erilaisia projektissa hyödynnettyjä testaustapoja lähdemateriaalin avulla. Mukana on esimerkkejä kunkin moduulin testitapauksista ja

tuloksista. Kuvaan tässä luvussa tapamme toteuttaa testaus ja esittelen dokumentointiin liittyviä huomioita. Pienien testauskokonaisuuksien, kuten toiminnallisuuksien ja yksikkötestauksen suorittaminen ja koko moduulin testauksen suorittaminen tähtäsi lopulta siihen, että kaikki ohjelmiston osat saatiin testattua. Tämän jälkeen oli mahdollista toteuttaa koko ohjelmistoa koskeva end-to-end -testaus tai myös eräänlainen tapa tehdä hyväksymistestaus. Altéa FM -ohjelmiston käyttöönotossa lopullinen hyväksymistestaus tehtiin ohjelmiston tuotantoversioiden testauksen avulla ja siellä suoritettu laaja testi käsitti kaikki normaaliin lennon prosessiin liittyvät toiminnallisuudet erilaisin poikkeuksin. Testaus tehtiin ensisijaisesti liiketoiminnan vaatimusten mukaisesti, eli ohjelmiston avulla piti pystyä suorittamaan koneen lähtöselvitykseen ja matkustajien lähtöselvitykseen liittyvät toimenpiteet.

Luvussa 5 avaan testausprosessin kehittämisen haasteita ja vertaan Altéa-projektin testauksen kypsyysastetta TMMi-mallin ja testausprosessin kypsyystasoon. Arvioin oman kokemukseni ja TMMi-malliin pohjautuen ja sen kypsyystasojen pohjalta, että mille tasolle Altéa-projektissa päästiin. Viralliseen arvioinnin toteuttamiseen tarvitaan sertifioidut arvioijat ja tämä voidaan Finnairilla toteuttaa, jos se nähdään tarpeelliseksi. Valitsin kyseisen mallin, sillä siinä on selkeästi kuvattu kunkin tason osalta ne vaatimukset, jotka edellytetään seuraavalla kypsyystasolle. Testausprosessiin projektissa vaikuttivat merkittävästi varsinainen testauksen toteutus päivittäisrutiineineen, testauksen mitattavuus ja raportointi ja myös testauksen hallinta ja aikataulutus. Testausprosessia kehitettiin projektin tarpeiden pohjalta ja jatkuvasti, kuten se projektin aikana tarpeelliseksi nähtiin. Testauksen ohjaus ja hallinta oli projektin loppuvaiheissa erittäin hyvin hoidettu. On hyvä huomioida, että testaus oli Altéa-projektissa vain yksi osa-alue muiden osien ohelle.

Viimeinen, eli luku 6 on varattu johtopäätöksille ja testausprosessin vaikutuksien arvioinnille Altéa-projektin näkökulmasta. Projektin ja testausprosessin opit, kehitystyön tulokset on esitelty tässä luvussa. Dokumentoinnin merkitys ja sen kehittäminen on avainasemassa testausprosessin parantamisessa. Dokumentointi takaa osaltaan uusien asioiden kehittämisen ja osaamisen jakamisen. Ylilaatuun ei kuitenkaan projektissa kannata sortua ja sama asia koskee testausprosessin kypsyystasoa. Prosessin tulisi olla organisaation vaatimusten mukainen ja laajuudeltaan hallittavissa. Prosessin osalta on tärkeää, että se on mitattavissa ja se on kuvattu, jotta sen osalta kaikille sitä toteuttaville tahoille on selkeää, miten eri seikat ja toimenpiteet vaikuttavat tähän kokonaisuuteen. Annan myös jatkokehittämiseen vinkkejä, joiden avulla projektin opit voidaan jalkauttaa tai ottaa toimintavoiksi Finnairilla.

2 OHJELMISTOTESTAUKSEN TAUSTAT JA TARKOITUS ALTÉA-PROJEKTISSA

Ohjelmistojen testaus on merkittävä tekijä ohjelmistohankkeen onnistumisessa. Haikala & Märijärvi (2002, s. 55-57) ovat pyrkineet osoittamaan, että laadunohjauksen keskeisimpiä tavoitteita on virheiden ennaltaehkäisy. Virheiden poistaminen on mahdollista testauksen avulla. Haikalan & Märijärven (2002, s. 57) mukaan testaus on hyvä toteuttaa mahdollisimman aikaisessa ohjelmistotuotannon elinkaaren vaiheessa. Näin voidaan välttyä testauksessa myöhemmin mahdollisesti ilmenevältä ja työläältä lisä- ja muutostyöltä (Boehm, 1987), joiden määrä lisääntyy Boehmin (1987, s. 4) ja Haikalan & Märijärven (2002, s. 57) mukaan hankkeen kehitystyön edetessä. Ohjelmistojen testaus vähentää erityisesti virheiden määrää tai epäloogisuuksia ohjelmiston käytössä, sekä vähentää ohjelmiston kehitys- ja ylläpitokustannuksia käyttöönoton jälkeen, jos testaus on suoritettu asianmukaisesti jo ennen käyttöönottovaihetta (Haikala & Märijärvi 2002, s. 40).

Tässä luvussa esittelen projektin, jossa olin mukana osana ohjelmiston testaus- ja käyttöönoton tiimiä. Ohjelmisto testattiin useilla eri menetelmillä ohjelmiston peruskäytön yksinkertaisista testitapauksista aina ohjelmiston kokonaisvaltaiseen testaukseen. Havaitsemiemme virheiden osalta ohjelmiston virheitä yleensä korjattiin, mutta yhtiölle kriittisissä asioissa päädyttiin uusien toiminnallisuuksien toteutukseen ja jopa niin pitkälle, että ohjelmaan tehtiin uusia ominaisuuksia puutteellisten määritysten takia.

Testauksessa pyrimme seuraamaan alihankkijan käyttämää ja tunnettua ohjelmiston testauksen V-mallia (Haikala & Märijärvi 2002, s. 287), jota voidaan helposti soveltaa koko ohjelmistotuotannon vesiputousmalliin (Haikala & Märijärvi 2002, s. 35-41). Näin saimme samalla kehitettyä testauskäytäntöjä Finnairille ja myös yleisesti ohjelmiston käyttöönottoa silmällä pitäen. Pyrin myöhemmin kuvaamaan tarkemmin, millaisia testaustarpeita projektin aikana oli ja miten testauksen suoritimme.

2.1 Projekti lyhyesti

Järjestelmän käyttöönottoprojekti toteutettiin Finnairilla vuosina 2007-2009 ja toimin itse projektissa yhtenä järjestelmäasiantuntijana. Projektissa testauksella oli merkittävä rooli hankkeen onnistumisen kannalta. Projektiin liittyi myös merkittävästi asennuksien suunnittelua ja näiden testaamista sekä kouluttamista. Varsinaiseen pääprojektiin kuului kaksi osaprojektia, joista kummassakin otettiin käyttöön Altéa-ohjelmistoperheen

ohjelmistosovellus (katso liite 1: Altéa-ohjelmistoperheen ohjelmistot ja niihin liittyvät avaintoiminnallisuudet). Ensimmäinen projekti liittyi Altéa Flight Management -ohjelmistoon ja toinen Altéa Customer Management -ohjelmistoon. Altéa Flight Management (Altéa FM) on lentokoneiden tasapainolaskelmien tekemiseksi suunniteltu ohjelmisto, jonka avulla järjestelmä automatiikan, asetettujen optimirajojen ja tiettyjen taustamuuttujien avulla pystyy laskemaan lentokoneelle tasapainolaskelmien edellyttämät arvot. Altéa Customer Management (Altéa CM) on ohjelmisto, jonka avulla lennoille tehdään asiakkaiden eli matkustajien hyväksyminen (Check in) ja vahvistaminen (Boarding). Altéa CM -ohjelmiston avulla hallinnoidaan myös erilaisia viestisovelluksia, yhteyttä varausjärjestelmiin ja matkatavaroiden hallintaa.

Projekteihin liittyi myös muita samanaikaisia hankkeita, kuten itsepalveluiden kehittäminen Altéa CM:n rinnalla ja itsepalveluohjelmiston testaus, sekä kaikkien sovellusten asennuksiin, tietoliikenteeseen ja tietoturvaan liittyvät toimenpiteet. Projekti sisälsi erittäin paljon dokumentointia, ohjeistuksen suunnittelua ja toteutusta, koulutuksen suunnittelua, koulutusta, hankkeen aikataulutuksen suunnittelua ja riskien kartoitusta. Näitä työvaiheita projektiryhmässä toteutettiin sekä tukifunktioiden avulla että projektin eri vastuuhenkilöiden avulla. Ilmailualalla asiaa vaikeuttaa myös se, että ohjelmisto pitää hyväksyttää ensin ilmailuviranomaisilla ja paikallisesti lentoasemilla, ennen kuin se voidaan ottaa käyttöön. Tämän lisäksi ohjelmiston asennus pitää vielä hyväksyttää suljettujen verkkojen omistajilla, kuten esimerkiksi SITA, RESA, ARINC. Koska Finnair toimii globaalisti, myös suljettuja verkkoja on reittiliikenteessä useita ja näiden eri toimittajien kanssa piti myös saada järjestelmä käyttöönotetuksi ympäri maailmaa. Hyväksymistestausta ei tee tällöin lentoyhtiö vaan ohjelmiston vastaanottaja ohjelmiston toimittajan kanssa. En käsittele näitä testausprosesseja syvällisemmin tässä lopputyössä. Oli haastavaa ja antoisaa olla mukana projekteissa, joissa myös suunniteltiin testauskäytäntöjä ja kokonaisvaltaisesti testauksen hallintaa tuleville ohjelmistoprojekteille.

2.2 Testaustarkoitus

Testauksen ja dokumentoinnin päätavoite lähti siitä, että Finnairin oli saatava tuotteet toimimaan Eurooppalaisen lentoyhtiön vaatimuksia vastaavaksi, jottei missään tilanteessa seuraa minkäänlaisia henkilövahinkoja. Toiseksi testauksen tarkoituksena oli myös estää mahdolliset taloudelliset vahingot, joita saattaisi syntyä, jos ohjelmiston käyttötapauksia ja säädettäviä esirajoituksia eri konetyyppien kanssa ei olisi testattu. Sovellukset ovat myös liiketoiminnan kannalta kriittisiä, koska ilman toimivaa ohjelmistoa tai tietoliikennettä merkittävä osa Finnairin ydinliiketoimintaa kärsisi tai ainakin hidastuisi jo puolen tunnin käyttökatkoksesta (Tietoviikko, 2012). Hankkeella oli kaupallisessa ryhmässä toteutuksen osalta koko organisaation tuki. Virheiden minimoiminen oli tärkeä osa projekteja ja testauksen merkitys korostui merkittävästi projektin aikana.

Hankkeen merkitys oli myös suuri myös siltä näkökannalta, että Finnair oli toinen lentoyhtiö, jonka kanssa Amadeus otti Altéa-järjestelmät käyttöön. Ensimmäinen lentoyhtiö oli Qantas. Oli erityisen tärkeää testata kaikki Finnairille tärkeät ominaisuudet, joita Qantas oli testannut aikaisemmin, mutta ennen kaikkea ne ominaisuudet, joita Qantaksen henkilöstö ei ollut vielä testannut tai ottanut käyttöön omassa käyttöönottoprojektissaan. Hankkeen ja projektien aikana Amadeus kehitti toimittajien heidän yhteistyökäytäntöjä lentoyhtiöiden kanssa. Tästä hyötyi sekä tilaaja, eli Finnair että toimittaja, eli Amadeus, jolle Altéa-projektien vetäminen ja järjestelmien uusiminen tulisi olemaan merkittävä kasvava kokonaisuus. Amadeukselle Altéa-projektien toimittaminen oli uutta ja näin oli myös mahdollista kehittää Altéa-toimituksiin liittyviä projektitoimia ja virstanpylväitä. Tämä näkyi projektin aikana Amadeuksen suunnalta tiettyjen projektinhallintatoimintojen tuomisena Finnairin projektitiimille normaalista poikkeavalla aikataululla tai tiettyjen vaiheiden priorisointiin liittyvien resurssien kohdentamisessa. Tulevaisuudessa oli määrä perustaa kaikkien ohjelmistotuotteen käyttöönottaneiden kanssa ohjelmistokehitystä ohjaava kehitysyöryhmä, mikä määrittelee ohjelmiston kehityssuuntia ja johon kuuluu toimittaja ja kaikkien ohjelmistoa käyttävien lentoyhtiöiden edustajat. Huomioitavaa oli erityisesti se, että Finnairilla ja Qantaksella oli kuitenkin erilaiset tarpeet ohjelmistojen käytölle, erityisesti Altéa FM:n osalta. Finnairilla otettiin käyttöön Altéa FM:n kaikki moduulit, kuten rahtimoduuli, matkatavaran käsittelyyn liittyvä moduuli ja polttoaineen syöttömoduuli. Qantaksella käytössä oli vain lennon tasapainolaskelmien tekemiseen liittyvä moduuli. Finnairilla tuotteen käyttöönoton yhteydessä prosesseja muutettiin myös merkittävästi, jotta ohjelmiston käyttö olisi mahdollisimman tehokasta. Qantaksella taas vanhat rakenteet pyrittiin organisaation sisällä säilyttämään, jotta ohjelmistohanke saataisiin menestyksekkäästi läpivietyä. Finnairilla kuitenkin Altéa-ohjelmistojen käyttöönoton yhteydessä myös tasapainolaskelmien tuottamiseen liittyviä vastuita muutettiin ja lentokoneiden suunnitelmien toteutus keskitettiin reittikohteista kahteen keskukseen maailmassa.

2.3 Ohjelmistoprojekti ja sen tarkoitus, sekä testattavat ohjelmistot

Projektin päätarkoitus oli testata, kouluttaa, asentaa ja käyttöönottaa Altéa Flight Management ja Customer Management -ohjelmistot kaikissa Finnairin reittikohteissa, lukuun ottamatta lomalentojen kohteita. Tämän lisäksi asennukset tehtiin Finnairilla vaadittuihin toimistoihin eri lentokentillä ja toimipisteissä (Back Office Locations). Jakelua varten asennukset tehtiin palvelun tarjoajien, Amadeuksen ja omalta osaltamme kohdennettujen resurssien kanssa yhteistyössä. Toimistoihin jakelu ja asennukset ja Suomen sisäiset asennukset Helsinkiä lukuun ottamatta tehtiin kootusti oman tiimimme kautta.

Itse olin mukana testaamassa lähinnä Altéa FM -järjestelmäprojektissa ohjelmiston tiettyjä moduuleita ja keskityin alihankkijan edustajien kanssa testauksen raportoinnin kehittämiseen. Ensimmäinen ongelma testauksen hallinnassa oli raportointikäytäntöjen kokemattomuus tai sopivan mallin käyttöönotto. Periaatteessa tarve tähän raportoinnin hallintaan ja edistymisen raportointiin lähti käyttöönoton tavoitteista. Tällä pyrittiin myös osoittamaan edistymisemme aikataulun suhteen. Pystyimme myös testausraportoinnin ja koordinoinnin avulla kohdentamaan resursseja sellaiseen projektin vaiheeseen, mikä näytti jäävän jälkeen muista kokonaisuuksista. Testauksen raportoinnista oli myös selkeää hyötyä pidemmän aikavälin resurssien suunnittelussa ja käytössä. Hetzelin (1998, s. 57) mukaan on syytä tarkentaa aika ajoin tavoitteita ja säätää joko aikataulua tai kohdentaa resurssien käyttöä oikeaan osa-alueeseen. Näin voidaan saavuttaa tavoitteet aikataulussa tai tehdä muutoksia aikataulun puitteissa tai resurssien mukaisesti. Myös Altéa FM -ohjelmiston testaamisessa oli useampaan otteeseen aikataulutettava tekemistä ja kohdennettava testauksen painopistettä sen mukaisesti, mikä oli ohjelmiston käyttöönoton kannalta ensisijaisen tärkeää ja priorisoida eri moduulien osalta testauspanosta. Lisäksi muutoshallinnan kautta tilattujen lisäominaisuuksien testaaminen piti resursoida ja aikatauluttaa, kun ne olivat valmiina ohjelmiston testausasennuksessa.

Projektissamme testattavana olleen Altéa-ohjelmistoperheen käyttöönotettavat kaksi tuotetta Finnarin Altéa-projektissa käsittivät seuraavat kaksi ohjelmistoa:

- Altéa Flight Management -ohjelmisto, joka on tarkoitettu lentokoneiden tasapainolaskelmien valmisteluun kaikki siihen liittyvät osa-alueet huomioon ottaen. Näitä osa-alueita ovat esimerkiksi matkustajat, rahti, matkatavara, catering ja lentopetroli.

- Altéa Customer Management -ohjelmisto, joka on tarkoitettu lähtöselvitystoimintaan lentokentällä (lähtöselvitys ja porttitoiminnot). Ohjelmiston avulla on myös mahdollista hallita häiriötilanteita ja ohjelmisto käyttöominaisuudet sisältävät esimerkiksi työnohjaukseen liittyviä toiminnallisuuksia, mahdollisuuden viestintään lentokentällä tai eri toimipaikkojen kesken ja matkustajien varausten hallinnan.

Liitteessä 1 on kuvattu Altéa-ohjelmistoperheen muut ohjelmistot ja liitteessä 2 on kuvattu Alta FM -ohjelmiston liitokset ja tietovirrat eri toimijoiden ja ohjelmistojen kesken.

2.4 Ohjelmistointegraatiot ja liitokset muista ohjelmistoista

Altéa FM -ohjelmisto koostuu useista eri ohjelmistomoduuleista, joilla hallitaan erityyppisiä lennon valmisteluun liittyviä tietoja ja jotka palvelevat eri tietoja tuottavia tahoja. Ohjelmistomoduulien kautta järjestelmään voidaan syöttää lennon tasapainolaskelmaa varten tarvittavat tiedot tai lähettää tiedot ohjelmistoon ennalta standardisoitua viestintäprotokollaa ja viestiformaattia käyttäen (liite 2: Altéa FM -ohjelmistoon tulevat ja siitä lähtevät tietovirrat eri toimijoilta). Eri toimijoilta, kuten lentorahdista ja cateringista, eli ateriapalvelusta, ja heidän ylläpitämistä ohjelmistoista tulee Altéa FM -ohjelmistoon tiedot standardisoidun viestiliikenteen kautta. Kolmannen osapuolen ohjelmistot eivät ole osana Altéa-ohjelmistoa, vaan niiden linkitys on toteutettu standardiin pohjautuvan viestiformaatin avulla tai manuaalisilla syötteillä. Integraatiotyö oli projektissa teknisten ihmisten osalta avainasemassa, jotta ohjelmistoon saatiin tarvittavat tiedot. Haasteena eri syötteiden ja ohjelmistointegraatioiden käyttöönotossa oli se, että Qantas ei näitä ottanut käyttöön, vaan ne käyttöönotettiin ensimmäisen kerran Finnairille. Tämän takia myöskään todellista systeemi-integraatiotestausta tai syötteiden toimivuutta ja niiden siirtoa Altéa FM -järjestelmään Finnairin käytössä olleista ohjelmistoista ei ollut testattu Amadeuksen Altéa-ohjelmistoon. Viestiformaattien selvittämiseen ja viestien läpimeno Altéa-järjestelmään vaati ensin testiympäristössä verifioinnin, mahdollisten korjausten siirron tuotantoympäristöön ja edelleen tuotantojärjestelmien välisen viestiliikenteen validoinnan. Joissakin tapauksissa vaadittiin edelleen ohjelmiston korjaustoimenpiteitä.

Seuraavissa alakohdissa on esitelty Altéa FM -sovelluksen tärkeimmät ohjelmistomoduulit, jotka muodostavat Altéa FM -ohjelmiston kokonaisuuden. Kaikkien moduulien käyttö ei ole välttämätöntä tai niitä voidaan myös käyttää eri tavalla eri lentoyhtiöissä. Kaikkien seuraavassa kuvattujen moduulien testaus ja käyttöönotto suoritettiin Finnairin Altéa-projektissa ja tietojen syöttö ja toimitus oli tiedon omistajan vastuulla. Tiedon omistajille tarjottiin kaksi vaihtoehtoa, eli viestipohjainen standardiviestin lähetysmahdollisuus Altéa FM -järjestelmään tai Altéa FM -järjestelmän asennus ja sen käyttö tietojen syöttämiseen. Tässä tapauksessa myös kummankin tavan käyttö on rinnakkain mahdollista. Liitteessä 2 on kuvattu ylätasolla Altéa FM -ohjelmistoon tulevat ja siitä lähtevät tietovirrat eri toimijoilta.

Rahtimoduuli ja rahdin vastuut

Rahdin vastuulla on Altéa FM -järjestelmään rahtitietojen syöttö manuaalisesti suoraan ohjelmistoon. Näin ohjelmistossa voidaan hyödyntää rahdin syöttämiä ennakkotietoja tasapainolaskelmissa. Lennon valmistelijalle on tärkeää tietää mahdollisimman aikaisin rahtivaraus, jotta he voivat suunnitella myös matkatavaroiden sijaintitiedot ohjelmiston avulla. Jos rahtia on liikaa, on lennon valmistelijan vastuulla ilmoittaa mahdollisista muutoksista vastaavalle rahtivirkailijalle. Vakioformaatissa lähetetty viesti tai viestit

tuottavat saman tiedon järjestelmään huomattavasti vähäisemmällä työllä, koska ne on mahdollista saada suoraan rahdin huolintaohjelmistosta Altéa-ohjelmistoon. Rahtimoduulin ja standardiviestin lähetys on myös mahdollista rinnakkain ja näin jommallakummalla tavalla toimitettu tieto päivittää edellisen tiedon uusin tiedoin. Viestin välittyessä Altéa FM -järjestelmään, se ajaa edelliset järjestelmään syötetyt tiedot tai edellisen viestin informaation yli. Varsinaisesti Altéa FM -ohjelmistoa ei tarvitse käyttää lainkaan, jos rahdissa pystytään tuottamaan viestit lentoa kohden rahdin huolintajärjestelmästä.

Rahdilla on käytössä myös erillinen viimeinen rahtitiedon viestiformaatti, joka vahvistaa suunniteltu rahtimäärä kunkin lennon osalta. Alustavalla suunnitelmalla ja lopullisella ja vahvistelulle rahtimäärälle ja sen viestille on erilainen koodi, joka on viestissä mukana. Näin lennon valmistelija tietää, kun viimeisin tieto on syötetty järjestelmään ja voi sen jälkeen antaa lennon kuormaukseen tarvittavat ohjeistukset.

Vaarallisten aineiden raportointi hoidetaan myös samassa viestissä liittyen muihin kuormaustietoihin. Rahtimoduulin käyttöönottoa tuettiin rahtihenkilöstön koulutuksin ennen kuin viestien toimitus järjestelmään oli varmistettu viestiformaatin avulla.

Catering, eli ateriapalvelu

Lennon ateriapalvelun, eli cateringin vastuulla on lähettää järjestelmään oikeat tiedot määrämuotoisena SITA-sanomana telexillä. Lähetettäviä tietoja ovat tarjoilukärryjen määrä, niiden sisältämä tuotteet, eli esimerkiksi ruokatarvikkeiden ja myyntituotteiden, sekä kärryjen painojen tiedot. Catering vastaa myös siitä, että tiedot täsmäävät koneeseen kuormatun kärrymäärän ja näiden painojen osalta, sillä ateriapalvelun toimittaja myös kuormaa ruoka- ja muut tarvikkeet lennoille. Tarvittaessa muuttuneet tiedot lähetetään Altéa-järjestelmään uudella viestillä, jolloin edellinen tieto pyyhkiytyy yli uudella tiedolla.

Kuormaustoiminnot ja kuormauksen kuittaaminen

Koneen kuormaus (rahti, matkatavara ja muu kuorma) raportoidaan kuormauksen toteutuman mukaisesti. Koneen lastaus kuitataan oikein ohjeen mukaisesti kuormatuksi lentoyhtiön ja ilmailuviranomaisten vaatiman ohjeistuksen mukaisesti. Lopullinen kuorman sijoittelu ja painot sekä kuorman kuormauksen järjestys vaikuttavat lentokoneen lähtöarvojen määrittelyyn ja lopullisesti lentokoneen tasapainolaskelmien arvoihin ja tasapainon määrittämiseen. Kuormaustoimintojen koneeseen kuormaamat artikkelit voidaan raportoida suoraan omasta Altéa FM -järjestelmän kuormauksen ohjelmistomoduulista tai myös viestin avulla määrämuotoisena SITA-sanomalla. SITA-sanomaa käytettäessä lopullisen kuorman sijoittamisen vastuu ja sen lopullinen kuittaus jää lennon valmistelijan vastuulle.

Kuormausmoduulissa on selkeästi näkyvissä ruumien sijainnit graafisessa käyttöliittymässä ja ruumien kohdalla on sijoitettuna suunniteltu kuormaus. Tarvittaessa

kuormausvastaava voi sitten siirtää kuormaa eri ruumien ja näiden sijaintien välillä tai sitten korjaamalla oikeat kuormauspaikat listattuun näyttöön. Periaatteessa kuormausvastaavalla on mahdollisuus muuttaa kuormausta sallituissa rajoissa, mutta tarvittaessa isommista muutoksista hänen on hyvä olla yhteydessä lennon valmistelijaan. Lopullinen kuittaus kuorman osalta voidaan tehdä vasta, kun lennon valmistelija on hyväksynyt mahdolliset kuormauksen tekemät muutokset. Tämä kuittaus voidaan kuitata Altéa FM -ohjelmiston kuormausmoduulista tai lähettämällä lennon valmistelijalle SITA-sanoma, jolla vahvistetaan lopullinen kuormaus.

Lentopetroli ja tankkaus

Lentopetrolin tankkausmäärä lasketaan ennalta eri reiteille omassa lentoyhtiön yksikössä, missä huomioidaan tankattavan lennon sääolosuhteet, lentoreitti ja varakentän sijainti. Lentopetrolin osalta tankattavan määrän tieto tulee Altéa FM -järjestelmään viestiliikenteenä alustavaksi pohja-arvoksi, jota käytetään hyväksi lentokoneen alustavaa tasapainolaskelmaa muodostettaessa. Koneen kapteeni tekee kuitenkin lopullisen päätöksen lentopetrolin määrästä ja lähettää oikean tiedon koneen viestijärjestelmästä ACARS-sanomana erilliselle ohjelmistolle, josta se edelleen siirtyy Altéa FM -ohjelmistoon tasapainolaskelman toteuttajalla, jos ennalta suunniteltu lentopetrolin määrä muuttuu. Kapteenille lopullisen lentopetrolin määrän raportoi koneen tankkaaja, joten tällöin viimeisin tieto lentopetrolista saadaan vasta noin 1-2 tuntia ennen koneen lähtöä Altéa FM -järjestelmään lopullisen kuormauslaskelman pohjaksi ja edelleen koneen kapteenille lähetettäväksi.

Lähtöselvitettävät matkustajat ja porttitoiminnot

Matkustajamäärät ja matkatavarat siirtyivät aluksi vanhasta Finnairin AY FIDO-lähtöselvitysjärjestelmästä määrämuotoisena viestinä kutakin matkustajaa kohden. Myöhemmässä vaiheessa, kun Altéa CM -ohjelmisto otettiin käyttöön, tiedot tulivat suoraan Altéa CM sovelluksesta Altéa FM -ohjelmistoon. Tässä oli ennen Altéa CM -ohjelmiston käyttöönottoa myös sellainen haaste, että matkustajatieto piti päivittää tasapainolaskelmiin alustavia laskelmia muodostettaessa aina käsin lennon varaustietojen avulla Altéa FM -ohjelmistoon. Matkustajien sijoittelu koneeseen piti hoitaa kotimaanlentojen osalta ennalta arvioiden koneen eri istuma-alueille.

Lopullinen matkustajien määrä tarkastettiin vasta, kun lähtöselvitys oli suljettu ja koneen lähtöportti oli avattu. Lentokentällä koneen portin avaaminen aiheutti sen, että lähtöselvitys sulkeutui tällöin automaattisesti. Erityisesti kotimaan vapaavalintaisten istumapaikkojen lennoilla koneen painojen jakaantuminen eri istuinalueiden osalta saatiin selville vasta kun lähtöportti oli suljettu ja viimeinen matkustaja oli istuutunut koneeseen. Matkustajan painot voitiin syöttää järjestelmään myös manuaalisesti, mikä vaati linkityksen katkaisua AY FIDO-järjestelmään kunkin lennon osalta. Tätä

toimintatapaa käytettiin poikkeustapauksissa, jos esimerkiksi lennolta siihen jo selvitetty matkustaja myöhästyi lennolta esimerkiksi edellisen lennon ollessa myöhässä.

Lennon valmistelu ja lentojen tasapainolaskelmien toteuttamiseen tarkoitettu moduuli

Kuormausohje, tasapainolaskelma ja lennon lähtöön liittyvät tarvittavat tiedot tuottaa uudessa mallissa keskitetty tasapainolaskelmiin erikoistunut yksikkö, eli Centralised Load Control (CLC). Ohjeet lähetetään koneen kapteenille, kuormaajille ja muille sidosryhmille viestiliikenteen avulla suoraan Altéa FM -ohjelmistosta koneen viestijärjestelmään ja tarvittaviin telex-osoitteisiin. Tarvittavat tiedot ovat myös saatavilla ohjelmistosta koko lennon suunnittelun ajan. Tiedot ohjelmistoon päivittyvät sitä mukaa, kun tiedot ohjelmistoon kussakin ohjelmistomoduuleissa päivitetään rahdista, lennon reitityksen suunnittelusta (lentopetrolin määrä), matkustajatiedot lähtöselvitysjärjestelmästä ja kuormaustiedot perustuen kuormaukseen. Tasapainolaskelman tiedot ja muut tarvittavat tiedot on myös saatavilla järjestelmästä myöhemmin lennon lähdön jälkeen ja tarvittaessa arkistointijärjestelmästä edelleen tietyn periodin ajan.

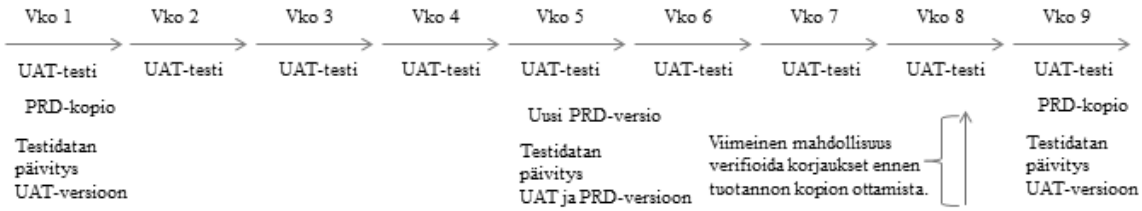
Altéa FM -ohjelmistoon asetettujen ennakkorajojen ja -asetusten avulla ohjelmisto laskee valmiiksi koneen etuosan ja takaosan painot (maksimi ja minimi toisiinsa nähden). Lisäksi lasketaan myös koneen kippirajat ja lentoon lähtöä varten asetettavat arvot koneen tasapainoon liittyen. Ohjelmisto myös kertoo optimaalisen laskeutumispainon ja kunkin koneen esiasetusten mukaisesti varoittaa, jos lennon lähtöpaino tai laskeutumispaino ylittää sille asetetut ennakkorajat. Vaikka tasapainolaskelmiin pohjautuva Altéa FM -sovellus otettiin käyttöön ja se osaa ehdottaa suoraan koneen nousuun liittyviä arvoja, on koneen kapteenilla aina mahdollisuus muuttaa ehdotettuja arvoja koneen lähtöhetkellä ennen nousukiitoa. Koneen kapteeni voi myös hakea Altéa FM -järjestelmästä tarvittavia tietoja milloin vain ennen lennon lähtöä ja näin valmistella lennon lähdön mahdollisimman tarkasti ennen viimeisimpiä tietoja.

2.5 Testausympäristöt ja tuotannon beta-version käyttö testauksessa

Projektin aikana Finnairilla oli käytössä sovelluksien testausympäristöt testilaboratoriossa, lokaalit testiasennukset järjestelmäasiantuntijoiden koneilla ja lisäksi erillinen koulutusympäristö, joka oli ajoittain melko epästabiili. Koulutusympäristö oli satunnaisesti esimerkiksi pois käytöstä, pohjadatan kopiointi järjestelmään ei toiminut kuten testiympäristössä ja sen lisäksi järjestelmän kaikki toiminnallisuudet eivät toimineet, kuin oli määritelty tai niissä oli puutteita. Projektissa oli koko projektin ajan myös käytössä tuotannon beta-versio ja vanhemmat tuotantoversiot. Tuotannon beta-

version osalta olimme hyvässä asemassa, sillä tässä ympäristössä oli projektin aikana vasta vain Qantaksen ja Finnairin tietoja. Toisaalta tämä myös rajoitti testaamista, koska jatkolentojen osalta lähtöselvittäminen niille oli haastavaa ja asiasta piti sopia erikseen Amadeuksen ja Qantaksen kanssa, jotta järjestelmään saatiin esimerkiksi muiden lentoyhtiöiden lentoja. Usein jatkolentotestitapaukset testattiinkin tekemällä järjestelmään Finnairin lentoja piilosarjoille ja kohteille, jota Finnair ei todellisuudessa lennä. Näille lennoille syötettiin jatkomatkustajia vanhan lähtöselvitysjärjestelmän kautta tai manuaalisesti suoraan Altéa FM -ohjelmistolla.

Testiohjelmistosta toimitettiin Amadeuksen toimesta joka viikko uusi versio, johon oli tuotu korjauksia, joista oli ilmoitettu virheiden raportointijärjestelmässä. Viikkorytmi muodostui standardinomaiseksi rutiiniksi ja alkoi yleensä aina uusien versiosovellusten lataamisella ja niiden asennuksella. Tuotantosovellus päivitettiin projektin aikana neljän viikon välein. Tähän ”kopio” hyväksytyistä toiminnallisuuksista otettiin testistä kolme viikkoa ennen tuotantosovelluksen päivityspäivää. Tämä toimintamalli asetti testaamiselle aikarajat, joiden osalta projektissa oli suunniteltava testausresurssien käyttö ja periaatteessa saavutettava tietyt testauspisteet ennen kunkin tuotantokopion uutta versiota. Jos jotakin asiaa ei saatu noin neljää viikkoa ennen tuotannon päivitystä testisovelluksessa verifioitua tai toiminnallisuutta hyväksyttyä, siirtyi tuotannon osalta päivitys seuraavat kahdeksan viikkoa eteenpäin projektin aikajanalla. Näiden eri ohjelmistoversioiden suhde on osoitettu kuvassa 2.1. Tuotantopäivityksen jälkeen keskityttiin noin kaksi viikkoa tuotantoversion (beta-versio) testaamiseen ja siellä toimiviksi todettujen ominaisuuksien osalta voitiin jatkaa perustestaamista testiversioiden avulla. Tärkeää oli ensin todentaa testisovelluksissa kaikki toiminnallisuudet niin hyvin kuin mahdollista ja validoida lopullinen toiminnallisuus toistamalla testitapaukset vielä kertaalleen tuotannossa.



Kuva 2.1. *Viikoittainen aikataulu, jossa on kuvattuna viikkotasolla ohjelmiston testin ja tuotannon versioiden päivityksien osalta suhteessa toisiinsa. Testi päivitettiin joka viikko ja tuotanto kerran neljässä viikossa. Testille (UAT-ohjelmistoon) tuotiin kerran neljässä viikossa testejä varten syöttämäämme dataa ja tuotannon beta-versioon (PRD-ohjelmistoon) pohjadata tuotiin kerran kahdeksassa viikossa.*

Sovellus itsessään oli Javalla toteutettu käyttöliittymä ja ohjelmisto, jonka kautta oltiin yhteydessä kahdennettujen tietoliikenneyhteyksien avulla Amadeuksen konesaliin Euroopassa, josta tieto oli hajautetusti käytettävissä ympäri maailmaa muissa

maailmalla sijaitsevissa konesaleissa. Amadeuksen palvelinsaleja sijaitsee kolmella mantereella, joten yhden palvelinsalin ollessa jostain syystä pois käytöstä pystytään hyödyntämään myös muita vastaavia saleja. Myös eri lentoyhtiöiden maantieteellisen sijainnin osalta palvelinsalien liikenne on hajautettu maantieteellisen sijainnin mukaisesti.

2.6 Muut testausvalmistelut testausta varten

Testaukseen käytettävän taustadatan syöttö kuului järjestelmäasiantuntijoiden vastuulle projektissa. Tämä oli testausta nopeuttavaa ja ennen kaikkea datan ennalta syöttäminen auttoi testauksen suorittamisessa, koordinoimisessa, aikataulun suunnittelemisessa sekä edelleen myös resurssien kohdentamisessa ja datan syötön vastuun jakamisessa. Käytössä oli joka neljäs viikko puhdas sarja lentoja testausta varten, jotka olimme suunnitelman ja ennakkoasetuksien osalta syöttäneet ohjelmistoa hyödyntävään tietokantaan. Uusien tietojen kopiointi tehtiin syöttämiemme tietojen pohjalta neljän viikon sykleissä. Neljän viikon välein pystyimme syöttämään testille sellaisia testitapauksiin liittyviä lähtötietoja, jotka testauksessa näimme tarpeelliseksi erilaisia testitapauksia varten (kuva 2.1.). Testidatan syötön koordinoiminen vaati projektissa myös datan syöttämisen vastuullisen henkilön työpanoksen kerran kuukaudessa. Testauksen lähtödatan kanssa oli myös omat ongelmansa, jotka periytyvät, jos datassa oli jokin virhe tai puutteellisia tietoja. Tämä virhe oli riesana seuraavat neljä viikkoa ja myös vaikeuttamassa testaamisen suorittamista.

Myös testitapausten ollessa monimutkaisia kunkin viikon lopulla alkuviikosta aloitettujen testitapausten loppuun vieminen oli toisinaan haastavaa, koska data palautui aina alkutilaan seuraavana viikonloppuna. Maanantaisin aloitettiin työskentely UAT-testillä puhtaalta pöydältä. Näin aina kuluvan neljän viikon syklin oli käytössä sama testidata ja viikoittain puhdasta testidataa, eli Altéa FM:n tapauksessa nämä olivat lentosarjoja erilaisilla konetyypeillä, matkustajamäärillä ja esimerkiksi polttoainemäärinä. Viikoittainen testidatan tyhjentäminen toisaalta toi tietyin väliajoin testitapausten suorittamiseksi paineet niiden viimeistelemiseksi kuluvan viikon aikana. Olemassa olevaa perusdataa oli mahdollista myös muokata testaukseen sopivien tarpeiden mukaisesti järjestelmässä ja näin saada poikkeavia tietoja testijärjestelmään. Toinen tapa, jota jouduimme toisinaan hyödyntämään, oli kullakin viikolla maanantaisin tehdyt alkuvalmistelut koko viikolle, jotta lähtökohdat testaukselle ylipäänsä olisivat kuluvan viikon ajan olemassa. Projektin aikana myös sattui muutaman kerran niin, että data oli jostain syystä käyttökelvotonta järjestelmässä, jolloin kaikki testitapauksien alkudata piti luoda aina kutakin testitapausta ajatelleen ennen testaamisen aloittamista.

Jokaiseen eri ympäristöön oli käytössä omat tunnuksensa, jotta eri ympäristöt eivät sekoittuisi keskenään. Myös kirjautuminen eri ohjelmistoympäristöön, esimerkiksi UAT (User Acceptance Testing Environment), PRD (Production) tai TRN (Training)

takasivat, että työskentely tapahtui kulloinkin oikeassa ympäristössä. Tunnuksien syöttäminen eri järjestelmiin Admin-ohjelmiston avulla ja tunnuksien huolto oli projektin ajan tukiorganisaation vastuulla. Tunnuksien osalta piti ottaa huomioon tunnuksien oikeuksien laajuudet eri käyttäjätasoille, tunnuksien hierarkkinen sijainti järjestelmätasolla ja tunnuksien tietojen toimittaminen oikeille tahoille. Projektin edetessä tunnuksien huolto ja hallinta siirrettiin operatiiviselle käyttötuelle. Näin saatiin jalkautettua tunnushuollon prosessit ja toimintatavat projektin aika niistä vastuussa olevalla taholle ja harjaannutettavaksi jo projektin aikana.

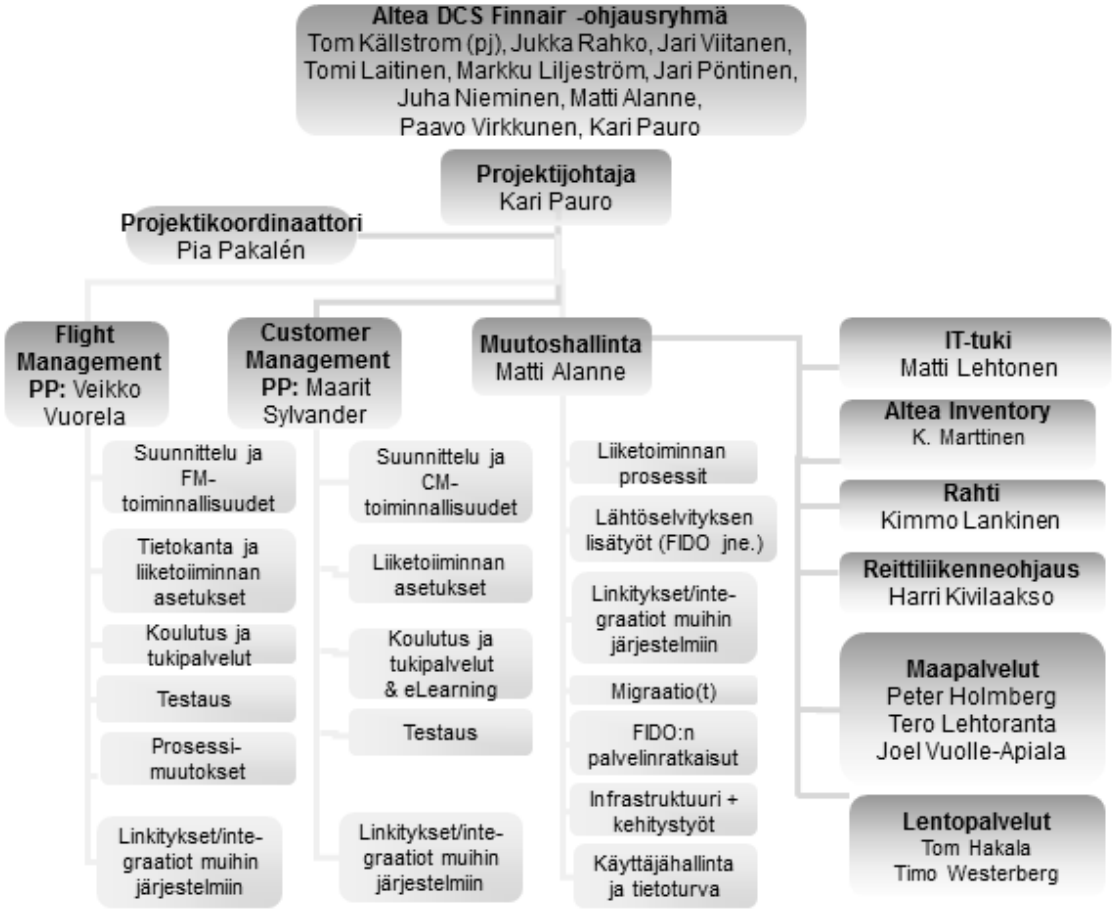
Integroitavien ohjelmistojen osalta asioita edistettiin sekä ohjelmistoista vastaavien teknisten tahojen että ohjelmistojen omistajaorganisaatioiden projektiin nimeämien vastuuhenkilöiden kanssa. Samalla suunniteltiin Altéa FM -ohjelmiston käyttöönottoon liittyen koko lentokoneen kääntämiseen liittyvien prosessien mahdollisia muutoksia ja aikataulutusta eri toimintojen osalta koneen kääntöön liittyen. Samalla päästiin ratkomaan myös mahdollisia ongelmia esimerkiksi viestiformaatin läpimenossa Amadeuksen asiantuntijoiden kanssa, jos viestissä oli jotain standardista poikkeavaa tai viesti ei esimerkiksi jostain syystä välittynyt Altéa FM -ohjelmistoon testillä tai tuotannossa. Syitä tähän saattoi olla esimerkiksi rahtiviestin välityksen osalta useita. Näitä olivat esimerkiksi väärä viestiformaatti, pilkkujen ja puolipisteiden käyttö väärässä paikassa, viestin lähettäjän osoitetietojen puuttuminen Amadeuksen järjestelmistä, jolloin viestiä ei hyväksytty edelleen järjestelmään toimitettavaksi ja vääränlaista informaatiota viestissä. Finnairilla havaittujen virheiden raportoiminen hoidettiin projektin aikana erillisellä Finnairin sisäiseen raportointiin käyttöönotetulla Bugzilla-ohjelmistolla, jonne tehtiin oma ryhmä Altéa-virheiden raportoimiseen. Näin saatiin kattavasti informaatio järjestelmään mahdollisten muiden toimittajien virheiden osalta ohjelmistointegraatioiden tasolla. Tarvittaessa, jos virhe ei näin ratkennut sisäisiä prosesseja tarkastelemalla ja lähdejärjestelmän viestin oikeellisuus oli todettu standardin mukaiseksi, koetettiin viestiä lähettää vielä toiseen järjestelmään verifiointia varten. Jos viesti meni läpi ja siitä oli todisteet eri järjestelmien välillä, oli Amadeuksen vuoro selvittää heidän järjestelmänsä rajoitteita ja mahdollisia ongelmia viestiformaatin osalta.

2.7 Projektiryhmä, vastuut ja projektin organisaatio

Altéa-projektissa oli mukana Finnair ja sen tytäryhtiöt, kaupallinen ryhmä tilaajana, Amadeus toimittajana, ITC Infotech alihankkijana, lukuisat maapalveluyhtiöt Finnairin alihankkijoina ja loppuasiakkaina, lentokenttien verkkojen omistajat asennuksien vastaanottajina. Projekti toteutettiin kahdessa vaiheessa. Ensimmäisessä Altéa FM -projektissa Finnairilta palkattujen asiantuntijoiden rooli oli lähinnä testaukseen liittyvien kokonaisuuksien edistäminen ja koulutustoiminnot (katso kuva 2.2.). Toisessa projektissa, eli Altéa CM -projektissa samat asiantuntijat keskittyivät taas enemmän projektin tukitoimintoihin eli käyttöönoton suunnitteluun, yhteistyöhön eri tahojen kanssa, prosessien ja käyttöönoton suunniteluun sekä asennustöihin. Altéa CM -

projektiin testausresursseiksi tuli kolme uutta henkilöä lähtöselvityspuolelta, joilla oli monen vuoden kokemus lähtöselvityksestä ja porttipalveluista. Altéa CM -projektin organisaatiokaavio liitteenä 4 selventää projektin laajuutta.

Testaustiimimme muodostui neljästä substanssiosaajasta Altéa FM -käyttöönotossa. Jokaisella testaajalla oli vastuullaan yhden ohjelmistointegraation testaaminen, sekä testattavaan ohjelmistomoduulin testitapausten suunnittelu, dokumentointi ja testauksen toteutus. Jokaisen testaajan piti tarkistaa jonkun toisen testaajan dokumentoimat testitapaukset. Mukana oli myös Intiasta testauskoordinaattori, joka edisti ohjelmiston perustestausta Suomen päässä intialaisten testaajien toimiessa Intiassa. Projektilla oli projektin johtaja ja kummallakin osa-projektilla oma projektipäällikkö. Nämä projektipäälliköt koordinoivat oman ohjelmistosovelluksen testausta ja käyttöönottoa Altéa CM:n tai Altéa FM:n osalta. Projektilla oli nimetty ohjausryhmä, joka valvoi projektin etenemistä, budjettia ja päätti mahdollisesta muutoshallinnasta.



Kuva 2.2. *Altéa FM -projektin organisaatiokaavio.*

Projektissa toimi projektin osana ohjelmistojen jakeluun ja asennukseen erikoistunut erillinen tiimi, johon itse liityin mukaan Altéa CM -ohjelmistoprojektin aikana. Osana kokonaisprojektia yhtenä sivuprojektina edistettiin myös lähtöselvitysautomaattien ohjelmistojen toteutusta, kehittämistä, asennuksia ja testaamista. Vanhojen Finnairin

ylläpitämien sovellusten sovelluskehittäjät työskentelivät myös tiiviisti mukana integraatiopuolella ja viestiliikennekehittämisessä, jotta tarvittavat vanhat ohjelmistosovellukset, kuten AY FIDO saatiin mukaan ja toimimaan yhdessä Altéa FM:n käyttöönoton aikana. Kokonaisuudessaan ja laajimmillaan projekti työllisti samaan aikaan yli 30 asiantuntijaa, joista Altéa FM -projektin osalta on lista liitteenä 3.

Koulutusvastuu oli jokaisella substanssiosaajalla, eli testaajalla. Käytännössä projektin elinkaaren aikana roolit hiukan muuttuivat, eli testaukseen keskittyneet tahot lähtivät suunnittelemaan koulutuksen laajuutta ja sisältöä ja valmistautumaan koulutuksen vetämiseen ja sen valmisteluihin. Jokaisella Altéa FM:n projektin osalta eri moduuleja testanneiden henkilöiden vastuulle jäi hänelle nimetyn moduulin koulutussuunnittelu, sillä testauksen perusteella moduulin toiminteet olivat hyvin hallussa. Kaikilta Altéa FM -projektiin osallistuneilta substanssiosaajilta edellytettiin koko järjestelmän tuntemusta, jotta heille oli mahdollista myös koko järjestelmän kouluttaminen ja toistensa tuuraaminen. Näin pystyttiin takamaan, että esimerkiksi sairastapauksissa oli nimetä varalle toinen kouluttaja.

2.8 Testauksen kustannukset

Testauksen kustannuksien osalta tehtiin tietyn tyyppistä laskennallista optimointia sen osalta, että verrattiin testaukseen käytettyä päivähintaa (resurssi- ja muut juoksevat kustannukset, sekä tietyt kiinteät kustannukset; vuokrat, laitteistot ja projektin hallinto) projektin valmistumisasteeseen. Näin voitiin peilata tarkemmin toteutuvia kustannuksia projektin aluksi budjetoituihin kokonaiskustannuksiin ja myös arvioida valmistumisasteen suhteen kustannuksia. Testauksen merkittävimmät kustannukset olivat siihen käytetyt henkilöresurssit, joita pyrittiin perustestaamisen osalta alihankinnan avulla pienentämään. Alihankinnan käyttö oli myös kahden muunkin seikan takia järkevää. Ensinnäkin projektin perustestaaminen vaati sellaisia testaustekniikoita ja myös automatisointia, joiden teettäminen alihankkijalla oli huomattavasti järkevämpää kuin Finnairin järjestelmäkehittäjille näiden asioiden kouluttaminen. Toisena syynä oli se, että ohjelmiston uusien versioiden testaaminen regressiomielessä oli myös suunniteltu projektin päätyttyä ja tulevaisuudessa alihankkijalle, joten heidän oli hyvä suunnitella testausautomaatiota tulevaa regressiotestaamista varten.

Projektissa kustannusten hallinnasta vastasi projektin vetäjä, joka asetti eri toiminnoille ja vaiheille budjettitavoitteet ja seurasi sekä käytettyä työaikaa että osaltaan tähän liittyviä muita kustannuksia, joista merkittävimpänä muita olivat matka- ja yöpymiskustannukset ja näihin liittyvät erilliset kustannukset kuten esimerkiksi päivärahat. Kustannuksia syntyi myös muutoshallinnan kautta sovelluksen osalta tarvittavien toiminnallisuuksien toteuttamisesta ohjelmistoon kehitystyönä. Kehitystöiden budjetoinnin suuntaviivat määriteltiin projektin ohjausryhmässä. Muita kustannustekijöitä olivat esimerkiksi ohjelmisto- ja laitekustannukset, testilaboratorio,

kolmansien osapuolien käyttämisestä seuranneet kustannukset, eli esimerkiksi tiettyjen ohjelmistoasennusten ja näiden testaamiseen käytetty aika, joka kului ohjelmiston suljettuihin ympäristöihin. Oman henkilöstön koulutuskustannukset, sekä erityisesti varsinaisten eri Altéa-ohjelmistojen loppukäyttäjien koulutuskokonaisuudet, olivat iso kustannus suhteessa koko projektin loppukustannuksiin.

Kustannusten seurannassa oli tärkeää niiden hallinta, koonti ja raportointi määräajoin ja tarvittaessa. Projektissa asetettiin myös ennalta määritellyt erilliset toleranssit, joiden sisällä tuli pysyä. Toisaalta aikataulua ja resurssien käyttö ja tätä mukaan kertyvät kustannukset kilpailivat periaatteessa toistensa kanssa ja näin jouduttiin priorisoimaan joko aikataulua tai ohjelmiston laatua ja siihen tarvittavia toiminnallisuuksia. Projektin edetessä myös budjettia tarkistettiin ja joidenkin osa-alueiden testausta siirrettiin alihankkijalle ja näin testausaikataulu pysyi perustestauksen osalta suunnitellussa aikataulussa. Muutoshallinnan kautta testaukseen tulleiden toiminnallisuuksien ja virheraportointien korjaukset viivästyttivät kuitenkin joltain osin Altéa FM -projektin varsinaista käyttöönottoaikataulua suunnittelusta.

2.9 Projektin ositus ja aikataulutus

Altéa-projekti oli vaiheistettu useaan osa-alueeseen ja varsinaisesti kahteen alaprojektiin kahden eri tuotteen käyttöönoton osalta. Tosin projektissa oli mukana muitakin laajempia erillisiä kokonaisuuksia, joita edistettiin ohjelmistojen käyttöönoton ohella. Näitä olivat aikaisemmin mainitsemani uusien lähtöselvitysautomaateissa sijaitsevien lähtöselvitysohjelmistojen (Self Service Check-in) käyttöönotto, asennukset ja kuormausohjeiden ja tasapainolaskelmien tuottaminen keskitetysti ja tähän liittyvät laajat prosessimuutokset lentokentillä ja eri operaatioihin liittyen.

Aikataulu ja projektin eteneminen oli korkealla tasolla jaettu seuraaviin työvaiheisiin Altéa FM -projektin osata, jotka osittain menivät päällekkäin toisten vaiheiden kanssa:

- Tammi- ja helmikuu 2007: projektin määritysdokumentaatioon tutustuminen, organisaation muodostuminen, vastuut projektin osalta ja uuteen emo-organisaatioon tutustuminen.
- Maalis-syyskuu 2007: Altéa FM -ohjelmiston testaaminen ja integraatioiden testaaminen.
- Kesä-syyskuu 2007: koulutuksen suunnittelu, valmistelu ja materiaalin tuottaminen.
- Heinä-joulukuu 2007: koulutukset eri osa-alueille ja toimijoille.
- Syyskuu 2007 - Maaliskuu 2008: Altéa FM -asennukset lentoasemille ja sovellusten testaaminen lopullisissa tuotantoympäristöissä.
- Tammikuu 2008: lopullisen tuotantosovelluksen käyttöönotto ja testaaminen tämän osalta (End-to-end -testaus ja Parallel-testaus).

- Helmikuu 2008: ensimmäinen lento Altéa FM -sovelluksella Finnairille Amsterdamista Helsinkiin.
- Helmi-toukokuu 2008: käyttöönoton tukeminen eri kohteissa Altéa FM:n osalta.
- Toukokuu 2008: kesäkohteiden koulutuksen sovellusten osalta niissä kohteissa, joiden osalta lennot sijoittivat kesäkaudelle.
- Kesä 2008: siirtyminen tukemaan Altéa CM -ohjelmiston käyttöönottoa ja siirtyminen uusiin rooleihin ja vastuisiin Altéa CM -projektin alkaessa.

Altéa CM:n projekti jakaantui pääpiirteittäin seuraavalla tavalla:

- Altéa CM -ohjelmiston eri moduulien ja toiminnallisuuksien testaus.
- Altéa CM - ja Altéa FM -ohjelmistojen testaus yhdessä ja muiden mahdollisten yhteyksien testaaminen eri sovellusten kesken.
- AY FIDO:sta siirtymisen suunnittelu Altéa CM -ohjelmistoon.
- Koulutuksen suunnittelu ja koulutusdatan ylläpitosuunnittelu.
- Sähköisen koulutusympäristön hyödyntäminen ennen varsinaisia koulutuksia maapalvelutyöntekijöille.
- Koulutukset useille sadoille maapalvelutekijöille ja koulutusjärjestelmien ylläpito.
- Asennukset ja asennustarkistukset kentillä ja toimistoissa.
- Varsinaisen testaukset kentällä ennen käyttöönottoa.
- Varsinainen käyttöönotto porrastetusti eri asemien osalta.

Kokonaisuudessaan projekti sisälsi kaksi erillistä ohjelmiston käyttöönottoa ja useita vaiheita ja testaus oli vain yksi osa-alue, joka korostui isossa roolissa Altéa-projektissa. Muita kokonaisuuksia projektissa olivat muun muassa koulutusvalmistelut ja koulutus, prosessien suunnittelu ja muutoksien jalkautus, ohjelmistoasennukset ja käyttöönotto ja sen tukeminen.

3 TESTAUKSEN TEORIAA ERILAISISSA PROJEKTIMALLEISSA JA TESTAUKSEN PROSESSIMALLI

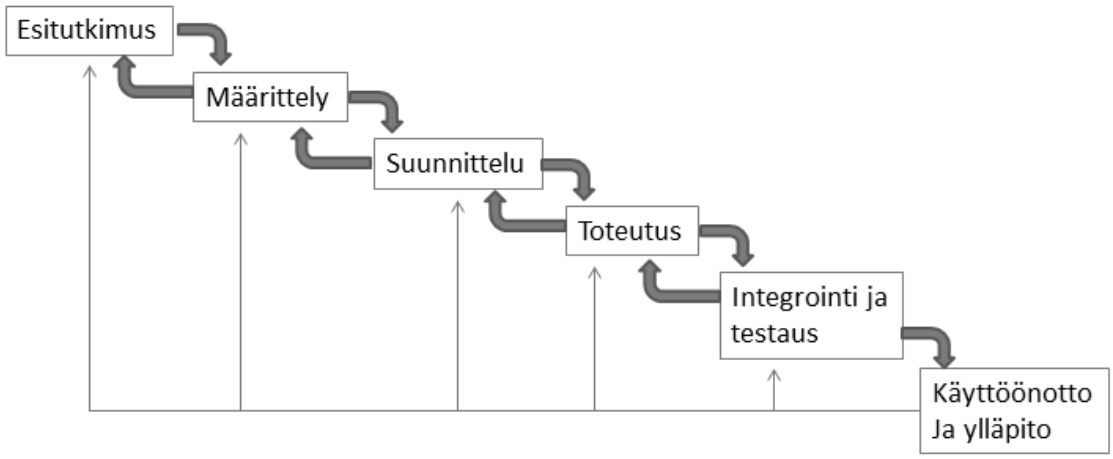
Esittelen tässä luvussa prosessipohjaisen ohjelmistokehitysmallin ja vertaan sitä ketterään Scrum-menetelmään. Avaan erityisesti testauksen osuutta kummassakin mallissa ja miten testaus suoritetaan suhteessa muihin ohjelmistokehityksen tehtäviin. Esittelen myös testauksen prosessimalleja ja tarkemmin TMMi:n prosessimallin teoreettiseen viitekehykseen peilaten ja esittelen sen arviointia TAMAR-kehikon avulla. Luvussa viisi peilaan testauksen prosessimallia Altéa-projektin työsuorituksiin ja siihen, miten prosessia hyödynnettiin ja testausta kehitettiin projektissa.

3.1 Prosessipohjainen toteutusmalli

Prosessipohjaisessa elinkaarimallissa testaus on yksi osa-alue muiden ohjelmistosuunnittelun osa-alueiden rinnalla. Tarkoitus on lähtökohtaisesti sijoittaa mukaan kaikki päävaiheet vaihejakomallin mukaisesti, mutta ottaa mukaan myös eri tukitoimintoja, jotka eivät varsinaisesti ole ohjelmistotuotteen elinkaareen liittyviä vaiheita, vaan tukevat ohjelmistotuotannon elinkaarta. Vesiputousmallissa vaihejako on jaoteltu Haikalan & Märijärven mukaan (2002, s. 35-41) mukaan seuraaviin vaiheisiin: määrittely, suunnittelu, toteutus ja yksikkötestaus, integrointi- ja järjestelmätestaus, käyttöönotto ja ylläpito. Immosen (2002) mukaan tähän kokonaisuuteen kuuluu vielä erilliset palvelut asiakkaalle. Näihin kaikkiin työvaiheisiin liittyy yhteiset tukitoiminnot, kuten tuotteen hallinta, muutosten hallinta, projektin hallinta sekä laadun ja riskien hallinta (Haikala & Märijärvi 2002; Immonen, 2002). Näiden avulla voidaan johtaa kokonaisvaltaiseen ohjelmistotuotteen hallintaan tai projektisalkun hallintaan. Haikalan & Märijärven (2002, s. 36) esittelemä vesiputousmalliin kuuluu näiden lisäksi vielä esitutkimus, joka tulee ennen määrittelyvaihetta (katso kuva 3.1.).

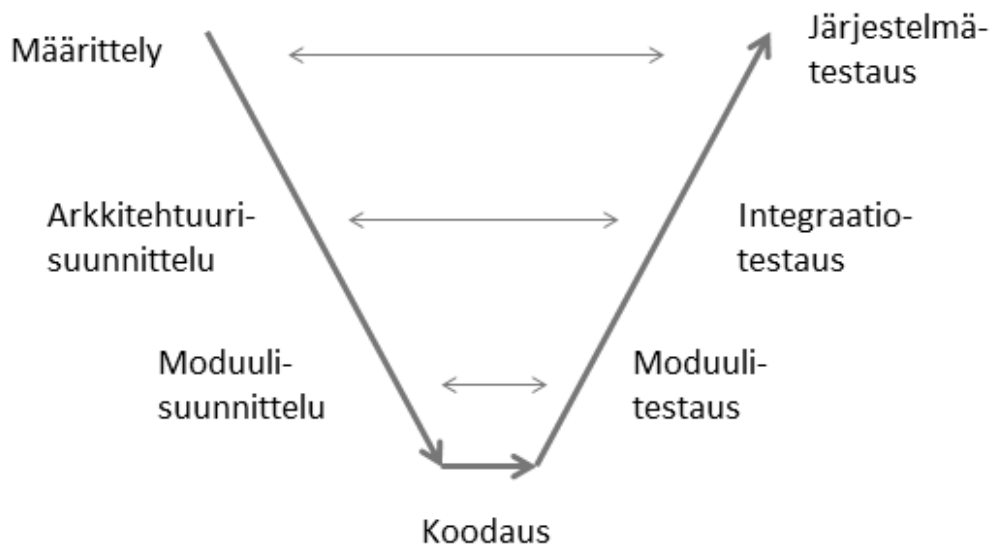
Esitutkimusvaiheessa kartoitetaan, että onko projekti tai ohjelmistokehityshanke mahdollista tai järkevää aloittaa ja millaisia kustannuksia tähän saattaa liittyä. Määrittelyvaiheessa kuvataan vaatimukset, jotka uuden toiminnallisuuden tai ohjelmiston tulee täyttää. On myös todella tärkeää kirjata asiat ylös ja esimerkiksi tuottaa määritysdokumentaatio. Suunnitteluvaiheessa tehdään teknistä suunnittelua sen osalta, että miten järjestelmä toteutetaan. Toteutusvaiheessa tehdään varsinainen koodaus ja rakennetaan järjestelmä tai ohjelmisto tai sen eri osat ja integrointi- ja testausvaiheessa ensin kootaan järjestelmä eri osista tai moduuleista toimivaksi

kokonaisuudeksi. Testaus vaiheessa tehdään erilaiset testit koko järjestelmän osalta, jotka voidaan toteuttaa esimerkiksi käyttötapausten avulla peilaten niitä esimerkiksi määrittelydokumentaatioon. Lopuksi hyväksytylle ohjelmistolle tehdään käyttöönotto suunnitelman mukaisesti, jota seuraa yleensä järjestelmän koulutukset ja tukitoiminnon. Tämän jälkeen siirrytään ylläpito- ja tukivaiheeseen, jolloin ohjelmiston ympäristöjä pidetään yllä, jotta sen käyttö on mahdollista. Jos ohjelmistoon halutaan tämän jälkeen muutoksia, voidaan käyttää muutoshallintaa, jossa muutokset ja niiden laajuus määritellään tai aloittaa esimerkiksi uusi projekti ohjelmiston uuden version kehittämistä ajatellen. (Haikala & Märijärvi 2002, s. 36-41.)



Kuva 3.1. *Vesiputousmalli (Haikala & Märijärvi 2002, s. 36).*

Vesiputousmallin osalta on johdettavissa testauksen V-malli (Haikala & Märijärvi 2002, s. 287), missä testauksen suunnittelussa otetaan huomioon tietyssä vaiheessa tietyn tyyppinen testauksen lähestymistapa. Tämä lähestymistapa kuvaa ohjelmiston elinkaaren osalta sen toteutusvaiheen edistymistä (katso kuva 3.2.). Määrittelyä peilataan järjestelmätestauksen avulla toteuttavaan kokonaisvaltaiseen testaukseen, mikä toteutetaan koko järjestelmän osalta. Moduulien suunnittelu ja niiden osa-alueet taas testataan moduulitestauksessa. Integrointitestauksessa testataan järjestelmän moduulien keskinäinen toiminta ohjelmistona, eli pyritään todentamaan, että järjestelmän moduulit toimivat rinnakkain toistensa kanssa. Järjestelmätestaus testaa ohjelmiston toiminnallisuuksia, eli miten ohjelmisto toimii vaatimuksia vastaan.



Kuva 3.2. *Testauksen V-malli (Haikala & Märijärvi 2002, s. 287).*

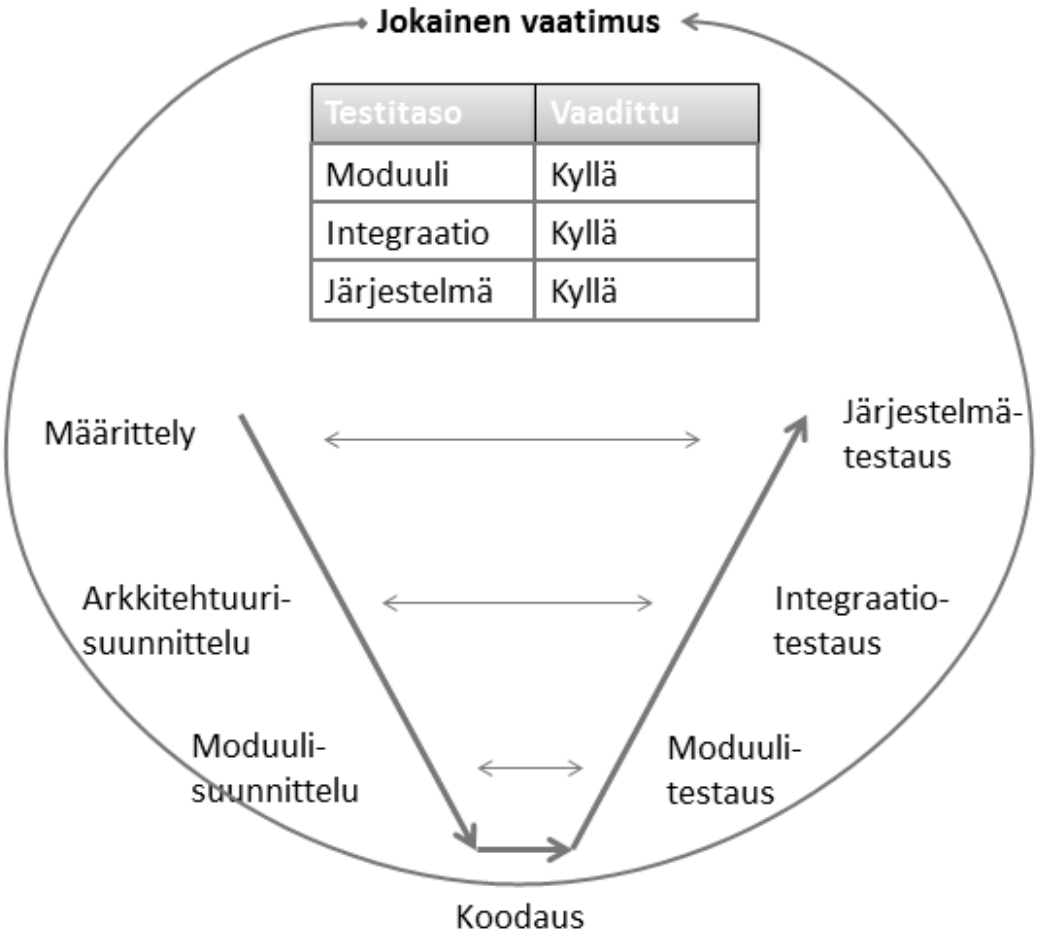
Altéa-projektissa ei varsinaisesti menty näin lähelle ohjelmiston koodin testausta, sillä projektissa moduuli tarkoitti yhtä ohjelmiston kokonaisuutta eli moduulia ja nämä jakaantuivat osa-alueiden osalta yhden testaajan vastuulle. Ohjelmiston integrointitestausta ei pidä sekoittaa kokonaisvaltaiseen systeemi-integraatiotestaukseen. Integroinnin osalta testauksessa yhdistellään eri moduulien ja järjestelmän osia suuremmiksi kokonaisuuksiksi (Haikala & Märijärvi 2002, s. 287). Systeemi-integraatiotestauksessa testataan kahden eri ohjelmiston keskinäinen toimivuus (Jääskeläinen, Katara & Vuorio 2012, s. 118).

Dokumentaation osalta projektissa oli hierarkkinen malli, joka kattoi kokonaisvaltaisen testaussuunnitelman (Master test plan) ja tämän alla oli V-mallin mukaisesti mukana erilaiset moduulien testaukseen liittyvä testausdokumentaatio. Dokumentaatioon kuului testauksen etenemisen raportointi, kokonaissysteemin testaaminen integraatioineen, Amadeuksen ja Finnairin kanssa yhdessä toteutettu end-to-end -testauksen testaussuunnitelma ja sen mukaisesti vielä erikseen toteutettu hyväksymistestauksen dokumentointi (Koomen & Martin 1999, s. 174-175). Dokumentaation tuottaminen vastuu oli eri henkilöillä projektiorganisaatioissa roolista riippuen. Esimerkiksi testausstrategian ylläpidosta vastasi koko hankkeen johtaja yhdessä ohjelmiston käyttöönottoprojektien projektipäälliköiden ja myös toimittajan projektipäällikön kanssa. Testaussuunnitelmien toteutuksen ja päivityksen vastuu oli projektipäälliköillä. Moduulikohtaisen testiaineiston tuottaminen oli järjestelmäasiantuntijoiden vastuulla, kuten myös ohjelmistointegraatiota tai viestiliikenteen avulla ohjelmistoon liittyvien ohjelmistojen dokumentaatio.

3.2 Scrum ohjelmistokehityksen mallina ja testaus Scrum-mallissa

Scrum-metodologiassa edetään lyhyin kehitysvaihein, jotka ovat esimerkiksi viikon pituisia jaksoja ja nämä sitten muodostavat pidemmän, eli esimerkiksi kuukauden pituisin ohjelmiston kehityksen elinkaaren (Tuomikoski 2009, s. 6). Scrum-malliin kuuluu backlock-lista, eli työlista, josta eri kehitystehtäviä otetaan kehittäjille kehitettäväksi sprintin, eli pyrähdyksen aikana. Sprintin kokonaisuudet ja laajuus, sekä näiden määrä tarkentuu suunnitteluvaiheessa, esimerkiksi yhteisten palaverien tai Scrum Masterin johdolla. Näin jatketaan kunnes kaikki vaadittavat ominaisuudet on saatu backlogista toteutettua esimerkiksi jonkin ohjelmiston osan tai koko ohjelmiston osalta. Scrumissa on avainasemassa kommunikaatio, eli päivittäin peilataan palavereissa tehdyt asiat ja etenemisen esteenä olevat seikat. Scrumissa viikko jakaantuukin päivittäiseen tekemiseen ja tavoitteiden asettamiseen päivittäisten työsuoritteiden osalta. Etenemisen esteenä olevat asiat Scrum-vetäjä pyrkii selvittämään ja poistamaan, jotta työskentely voi jatkua. Sprintit osien valmistuessa ja eri kokonaisuuksien toteutuksen jälkeen on valmiina osa-kokonaisuus tai ohjelmisto. Lisäksi on mahdollista ja usein syytäkin pitää sprintin tarkasteluun liittyvä palaveri sprintin lopussa, jolloin käydään läpi kokonaisuus, jota sprintissä on saatu valmiiksi, eli esitellään tulokset esimerkiksi sidosryhmille tai tilaajille.

Scrumissa oleellista on, että jokainen ohjelmiston kehittämistyöhön osallistuva on sitoutunut yhteisten tulosten saavuttamiseksi, mutta se sisältää myös tiettyjen roolien läsnäoloa onnistuakseen (Szalvay 2009). Erityisesti tämä näkyy testaamisen organisoinnissa Scrum-mallin mukaisten ohjelmistokehitykseen tähtäävässä projektissa, jossa kokonaisvastuu testauksesta on sekä kehittäjällä että testaajalla. Scrumiin voidaan soveltaa Evansin (2008, s. 13) mukaan V-mallin tyyppistä lähestymistapaa jokaisen vaatimuksen testaamiseen, kuten kuva 3.3. osoittaa. Näin kukin vaatimuksen mukainen komponentti, koko systeemi ja mahdollisesti integraatioiden testaaminen aloitetaan alhaalta ylöspäin ohjelmistokoodin testaamisella jo ensimmäisenä kehityspäivänä ja lopulta voidaan myös suorittaa hyväksymistestaus kunkin erillisen vaatimuksen suhteen (Evans 2008, s. 17). Tämä voidaan saavuttaa siten, että jo kehittäjät testaavat oman toteuttamansa koodin toiminnallisuuden ja eheyden, jolloin testaajalle jää testattavaksi varsinaisen vaatimuksen kriteerien mukainen toiminnallinen puoli. Jaettuun vastuuseen kuuluu osana myös palautteen antaminen, jolloin korjauksien tekeminen vaatimuksien nähden on mahdollista asiakaspalautteen ja testaajan raportin pohjalta.



Kuva 3.3. *V-mallin implementointi Scrumiin Evansin (2008, s. 13) esittämän mukaisesti.*

Scrumissa on kolme tärkeää roolia, jotka ovat tuotteen omistaja, Scrum-vetäjä ja tiimin jäsenet (Szalvay 2009). Tätä kokonaisuutta voidaan tarpeen tulleen laajentaa muilla erilaisilla rooleilla. Osa rooleista voi olla myös päällekkäisiä, eli esimerkiksi Scrum-vetäjä voi toimia myös tiimin jäsenenä ja edistää ohjelmiston ohjelmistokehitystä. Tuotteen omistaja on vastuussa siitä, että hän välittää kehittäjätiimille esimerkiksi asiakaan vaatimukset ja tavoitteet sen osalta, mitä tuotteelta vaaditaan. Hänellä pitää olla selkeä kuva asiakkaan vaatimuksista, jolloin on mahdollista dokumentoida myös hyväksymiskriteerit, joiden avulla testaaja tai vastaava taho voi osoittaa, että toteutus vastaa vaatimuksia. Tuotteen omistajan roolissa tulee myös hallita mahdollisen asiakkaan kehitysehdotukset, kehut ja negatiivinen palaute. Usein tuotteen omistaja on asiakkaan edustaja, mutta toisaalta ko. henkilö voi olla myös Scrum-vetäjä. Roolien sekoittaminen on kuitenkin erittäin haasteellista ja suuri syy Scrum-mallin epäonnistumiseen. Szalvayn (2009) mukaan Scrum-vetäjän tulee toimia sekä auktoriteettina kehitystiimille että vastuullisena asiakkaan edustajana vaatimusten esittämiseksi, joten tasapaino näiden kahden roolin välillä vaatii harjoitusta. Toisaalta tuotteen omistajalla on ohjeistamis- ja johtamisvastuu kehitystiimille, eli on toimittava rakentavassa ja muita tukevassa roolissa, mutta ohjata myös tiimin toimintaa tavoitteita asettamalla.

Scrum-vetäjä (ScrumMaster) vastaa tiimin päivittäisen ja viikoittaisen toiminnan ja tiimin ohjaamisesta. Hän on vastuussa tuotteen omistajalle ja tiimille, mutta merkittävä ero työn johtamisessa on, että hän ei hallinnoi tiimiä vaan pyrkii poistamaan töitä estävät esteet ja ratkaisemaan mahdolliset haasteet, jotta töitä voidaan jatkaa (Szalvay 2009). Hänen vastuulla on myös organisoida työt niin, että tehtävät tulevat mahdollisimman hyvin tehtyä.

Tiimin jäsenen vastuulla on varsinaisten työsuoritteiden tekeminen. Tiimin koko voi vaihdella, mutta siitä pitäisi löytyä vähintään seuraavat roolit: ohjelmistokehittäjiä, ohjelmistoarkkitehti, tietokantaosaaja, laatuvastaava, testaaja ja mahdollisesti käyttöliittymävastaava. Tietenkin roolit voivat olla myös päällekkäisiä, jolloin yksi henkilö vastaa useammasta kokonaisuudesta sprintin aikana. Jokaisen pyrähdyksen aikana tiimi itse päättää, miten siihen määritellyt työt saadaan suoritettua ja mahdollisesti sen, että mitä jätetään esimerkiksi seuraavaan sprinttiin tai kokonaan tekemättä. Vapausaste tiimin sisällä on suuri, mutta toisaalta jokaisella tiimin jäsenellä on vastuu sprintin etenemisestä ja lopputuloksesta. Esimerkiksi testauksen osalta voidaan toimia siten, että kun vaadittu kokonaisuus tulee valmiiksi, niin tuotteen omistaja tai testaaja kirjoittaa testitapauksen käyttötapauksen perusteella perustuen vaatimuksiin tai määritykseen ja sen jälkeen testaa toiminnallisuuden. Jos testi ei mene läpi, niin tehtävä palautetaan sen toteuttajalle korjattavaksi ja edelleen arkkitehdille uudelleen suunniteltavaksi tarvittaessa.

3.3 Testausprosessien kehittäminen ja kehitysmallit

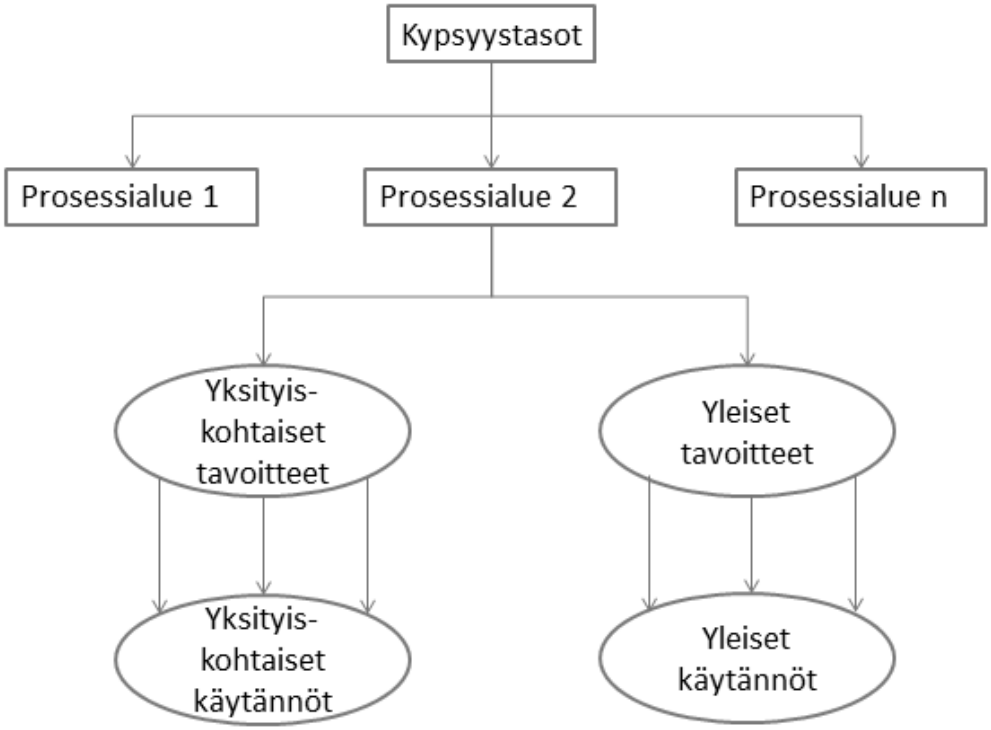
Maailmalla on useita prosessin kehittämiseen liittyviä malleja ja myös testauksen kehittämiseen soveltuvaa prosessimallia. Testauksen dokumentaatiota ja varsinaista testausta (verifiointi ja validointi) varten ovat myös IEEE-standardit, kuten IEEE-829 ja IEEE-1012. Ensimmäinen on testaukseen liittyvä standardi ja toinen syventää verifioinnin ja validoinnin käytäntöjä. Näiden standardien pohjalta testauksen lähtökohtia on hyvä soveltaa esimerkiksi erilaisissa projekteissa ja testaukseen erikoistuneissa yrityksissä. Uusimpia malleja on TMMi (Test Maturity Model Integration), jonka avulla testauksen prosessi voidaan arvioida asteikolla yhdestä viiteen, joka on korkein taso TMMi-mallissa. Toinen malli on esimerkiksi 90-luvulla Illinois Institute of Technology:ssa kehitetty TMM (Testing Maturity Model). Malli on testauksen kehittämisen perusmalli, jota voi soveltaa CMM:n (Capability Maturity Model) rinnalla. TIM (Test Improvement Model) julkaistiin Ruotsissa kehitetyn mallin mukaan samoihin aikoihin ja siinä kehitystasoja on vain neljä kappaletta. TIM:n hyvä puoli on sen keveys ja mallia voi hyvin käyttää esimerkiksi pienemmissä projekteissa mittaamassa projektin testauksen kypsyyttä. TMMi-malli on TMMi Foundationin malli, joka on kehitetty taas 2000-luvulla. Malli on kattava ja sitä myöten myös raskas, joten sen soveltuva käyttö on myös mahdollista. Prosessien kehittämiseen on TMMi:n rinnalla hyvä tällöin soveltaa esimerkiksi Capability Maturity Model Integration -mallia (CMMI). TPI-

malli (Test Process Improvement), joka on kaupalliseen tarkoitukseen kehitetty malli, olisi myös ollut hyvä arviointiväline prosessin ja testauksen kypsyyteen. TPI:ssä on itsessään 13 avainaluetta, joiden pohjalta testausprosessin kypsyyttä arvioidaan näiden avainalueiden kypsyyttä ja arviointi voidaan myös suorittaa avainalueiden osalta itsenäisesti. TPI-mallissa arvioinnin yhteenvetoon tarvitaan sen verran enemmän kokemusta matriisin avulla suoritettuna, että päädyin käyttämään TMM:n pohjalta kehitettyä TMMi-mallia. (Arovaara 2008, s. 13-21; 27-79.)

Altéa-projektissa testauksen kehittymistä arvioin erityisesti TMMi-mallin avulla ja arvioin projektissa kehittymisen tasoa suhteessa lähtötasoon ja TMMi-mallin kypsyystasoihin. Myös alihankkijamme testauksen prosessin monitorointi tapahtui projektissa sovitun mallin mukaisesti. Arvioimani lopputulos on prosessimallin mukaan vain viitteellinen ja ohjeistava, sillä varsinaiseen arviointiin tarvitaan Foundation-tason henkilö suorittamaan arviointi organisaatiossa. Tämä ei kuitenkaan projektin tai Finnairin kannalta ollut niin tärkeä seikka, että sellainen olisi kannattanut tehdä.

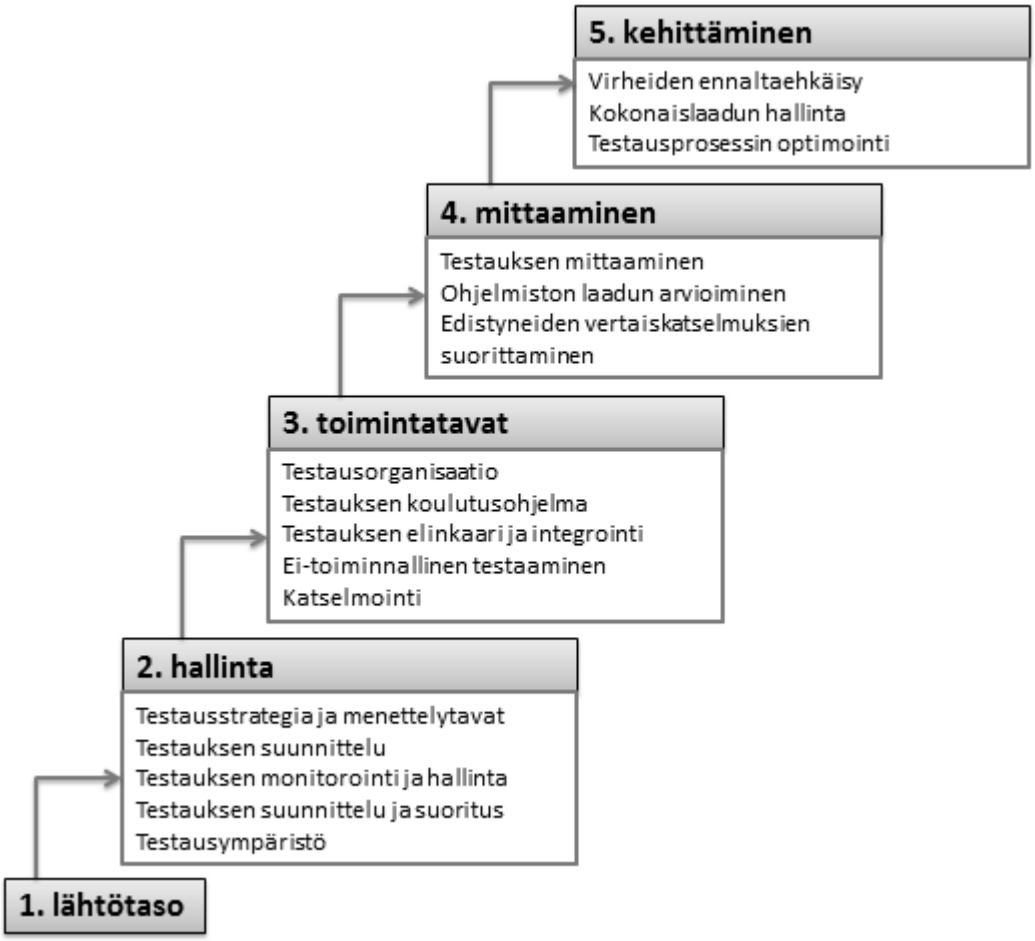
3.3.1 Testauksen prosessimalli TMMi:n mukaisesti

TMMi:ssä kypsyystasoja on viisi kuten jo aikaisemmin mainitussa TMM:ssä. Tässä kypsyystasot jakautuvat erillisiin prosessialueisiin. Kypsyystasot ja niiden prosessialueet on esitelty kuvassa 3.5. Prosessialueilla on van Veenendaalin (2012, s. 14) mukaan yleiset tavoitteet ja yksityiskohtaiset asetetut tavoitteet. Yksityiskohtaiset tavoitteet ovat niitä asioita, jotka pitää prosessin kullakin tasolla saavuttaa kunkin yksityiskohtaisen prosessin osakokonaisuuden arvioinnin osalta, jotta TMMi-tason vaatimukset täyttyvät. Vaaditut yksityiskohtaiset asiat tai käytännöt (Required Components) pitää siis aina olla kunnossa, jotta kunkin kypsyystason mukaiset ominaisuudet voidaan saavuttaa prosessikohtaisesti (van Veenendaal 2012, s. 13). Kullakin tasolla esiintyy myös odotettuja yleisiä käytäntöjä (Expected Components), mitkä liittyvät testauksen prosessin kehittämiseen ja testauksen käytäntöjen implementointiin organisaatiossa. Yleiset käytännöt voivat olla yhteisiä useammallekin prosessin osa-alueelle kypsyystasolla. Nämä ovat asioita tai toimintatapoja, joiden avulla prosessin eri osat tai toiminnot tuodaan organisaatioissa käytännöiksi. Kolmas asia, joka organisaatiossa tukee prosessia, on informatiivinen komponentti (Informative Components), joka tukee organisaatiota saavuttamaan vaaditut ja odotetut asiat organisaatiossa. Näitä informatiivisia välineitä voivat olla esimerkiksi muistiinpanot ja prosessikuvaukset. Seuraavalla sivulla oleva kuva 3.4. selventää TMMi:n rakennetta kypsyystasojen ja prosessialueen tavoitteiden osalta.



Kuva 3.4. *TMMi:n rakenne ja komponentit, eli yksityiskohtaiset tavoitteet ja yleiset tavoitteet (van Veenendaal 2012, s. 14).*

Tasolta seuraavalle kypsyystasolle pääsee, kun yksityiskohtaiset tavoitteet ja kullekin prosessialueelle tasolle asetetut yleiset tavoitteet on saavutettu. Tämä prosessin ylemmän tason vaatimusten täyttäminen edellyttää, että korkeammalla tasolla on myös kaikki alimman tason prosessien osakokonaisuuksien vaatimukset saavutettu tietyllä tarkkuudella (van Veenendaal 2012, s. 9). TAMAR-kehikon (TMMi Assessment Method Application Requirements) avulla on mahdollista arvioida TMMi:n mallin mukaiset toiminnallisuudet ja testausprosessin kypsyys organisaatiossa. TAMAR-kehikon (Goslin 2009, s. 1-26) dokumentissa on myös ohjeistettu, että miten arviointi tulee suorittaa ja dokumentoida. Arvioinnin suorittamisessa käytännössä käydään läpi kaikki käytännöt ja testaukseen liittyvät asiat, jotta prosessialueiden osalta ja kustakin kypsyystasosta saadaan selkeä käsitys.



Kuva 3.5. *TMMi:n prosessialueen kypsyystasot (van Veenendaal 2012, s. 9).*

Seuraavissa alakohdissa on esitelty eri tasojen kuvaukset lähteiden Veenendaalin (2012, s. 7-202) ja Arovaaran (2008, s. 69-75) mukaisesti.

3.3.2 Taso 1: lähtötaso

Testausprosessilla ei ole ensimmäisen tason mukaisia toiminnallisuuksia. Ei ole olemassa testikäytäntöjä tai testausstrategiaa. Suunnitelmallisuutta tai ohjausta ei tämän perusteella myöskään voida tunnistaa. Monitorointia, valvontaa ja kontrollointia testitapausten suhteen ei ole olemassa tai tunnistettu. Testitapauksille ei ole tiettyä ennalta määriteltyä dokumentaatiota tai toteutustapaa ja testiympäristöjen käytölle ei ole edellytyksiä tai niitä ei ole lainkaan. Tässä tapauksessa testejä tehdään yleensä suunnittelemattomasti ja dokumentoimatta tai dokumentoinnissa on puutteita, eikä sitä valvota (Arovaara 2008, s. 69). Tällä tasolla vaikuttavat testaajien omat käytännöt ja näiden noudattamista ei ohjata organisaation ohjauksella (Cannegieter & van Veenendaal 2013, s. 7). Tyypillisiä testaukseen liittyviä ongelmia ovat tällä tasolla sitoutumattomuus testaukseen tai testauskäytäntöjen hylkääminen ongelmatilanteissa (van Veenendaal 2012, s. 10).

3.3.3 Taso 2: hallinta

Tasolla kaksi on jo useampi prosessin alue arvioinnissa, joiden suhteen tulee olla osoitettavissa, että ainakin kyseiset asiat ovat kunnossa, jotta voidaan päästä edelleen TMMi:n tasolle kolme. Tason perusominaisuuksiin kuuluvat testausstrategian toteuttaminen, esimerkiksi dokumentoimalla se ja liittämällä samaan dokumenttiin testaustavat ja mahdollisuuksien mukaisesti edelleen testauksen toteutukseen liittyvien dokumenttien rakenne ja käyttötarkoitukset. Edelleen testausstrategiaa ja testaussuunnitelmaa tulee päivittää tarpeen mukaan. Näin saadaan karkea kuva testauksen suunnittelun ja toteutuksen pohjaksi.

Avainasemassa on testauksen suunnittelu ja testaussuunnitelman toteutus, eli mitä, milloin ja miten testataan. Organisaatiossa on käytössä testauksen menettelytavat ja dokumentoitu ja toteutettu testausstrategia (van Veenendaal 2012, s. 7). Tärkeä lähtökohta TMMi:n tasolla kaksi on periaate varmistaa, että testattava tuote vastaa vaatimuksia ja ohjelmistomäärityksen mukaista toteutusta (Cannegieter & van Veenendaal 2013, s. 7). Toisaalta esimerkiksi ohjelmiston toiminnallisuuksien osalta vaatimukset on täytettävä ennalta asetettujen tavoitteiden osalta. Testauksen toteutus suunnitellaan yleensä niistä lähtökohdista, että lopullisesta käyttöönotettavasta tuotteesta saadaan vaatimuksia vastaava tuote. Jos tässä tapauksessa testaus suoritetaan vasta kun ohjelmisto on jo luovutusta vaille valmis, täytyy ohjelmistoon tarvittavien uusien ominaisuuksien toteutus tehdä muutoshallinnan avulla. Tämä on Cannegieter & van Veenendaalin (2013, s. 7) mukaan yleisimpiä ongelmia edelleen tasolla kaksi. Testaussuunnitelma voi yksinkertaisimmillaan sisältää kunkin moduulin testauksen vastuuhenkilöt, listan tunnistetuista testitapauksista ja aikataulutuksen testaukselle. Testaussuunnitelman laajuus riippuu sekä testattavan kohteen laajuudesta että projektin laajuudesta ja prosessin tuomista vaatimuksista sitä kohtaan. Testaussuunnitelma ottaa myös tarkemmin kantaa siihen, että millaisilla tekniikoilla testaus suoritetaan (Arovaara 2008, s. 70). Testaussuunnitelmaan voidaan sisällyttää esimerkiksi seuraavia kokonaisuuksia: testausaikataulu, resurssit, tunnistetut sidosryhmät ja heidän tavoitteet sekä riskit (van Veenendaal 2012, s. 39-41). Lisäksi IEEE-standardien mukaisissa dokumenteissa on tietyntyyppisiä suosituksia sen osalta, mitä testaussuunnitelmassa tulisi olla mukana.

Tason kaksi TMMi-prosessin kannalta tarkastellaan van Veenendaalin (2012, s. 10; 24-78) mukaan seuraavia alueita:

- **Testausstrategia ja menettelytavat**, eli yksityiskohtaiset käytännöt sisältävät seuraavat osa-alueet: testauspolitiikka, testausstrategia, testauksen mittauksen indikaattorit.

- **Testauksen suunnittelu**, missä yksityiskohtaiset käytännöt vaativat seuraavat kokonaisuudet: riskit ja niiden hallinta, testaus ja testauksen

rajaus, testauksen suunnittelu ja toteutus, testauksen kustannusten hallinta, testaussuunnitelma ja sen katselmointi sekä töiden ohjaus.

- **Testauksen monitorointi ja hallinta.** Yksityiskohtaiset käytännöt tässä prosessin osa-alueessa koostuvat testauksen seurannasta suhteessa testaussuunnitelmaan, tuotteen tai testauksen laadun seurantaa suhteessa testaussuunnitelmaan raportoinnin ja katselmuksien avulla, sekä eri aktiviteettien hallinta, jotta testaus on kokonaisuudessaan mahdollista saattaa päätökseen.

- **Testauksen suunnittelu ja suoritus** sisältää yksityiskohtaisten käytäntöjen osalta seuraavat kokonaisuudet: testauksen analysointi ja suunnittelu testaustekniikoita hyödyntäen, testauksen läpivienti, testauksen suorittaminen, testauksen päättämiseksi tehtävät toimenpiteet ja tuloksien dokumentointi.

- **Testausympäristön** osalta yksityiskohtaisten käytäntöjen kokonaisuuteen kuuluu testausympäristön vaatimuksien saavutettavuus ja sen pystyttäminen, testausympäristön hallinta ja kehittäminen.

Testauksen monitorointi, aikataulutus ja hallinta kuvataan myös testaussuunnitelmassa. Testaussuunnitelma saattaa sisältää esimerkiksi raportointikäytännöt viikko- tai kuukausitasolla, mitattavat kohteet esimerkiksi yhdessä tai useammassa ohjelmistomoduulissa ja tarvittavat toimenpiteet viikkotason ohjaukseen jos aikataulu on kriittinen ja määritellyt riskitasot toteutuvat. Näiden estämiseksi esimerkiksi erilaiset toimenpiteet voidaan kuvata testaussuunnitelmassa. Tasolla kaksi tärkeintä on testauksen ohjaamisen ja hallinnan sekä johtamisen toteutus yrityksen tai projektin tasolla.

TMMi:n tasolla 2 yrityksellä tai projektissa tulee olla myös erillinen testausympäristö ja dokumentoituna tämän ympäristön rajoitteet, päivityssykli ja muut huomioon otettavat seikat. Testausympäristön hallintaan kuuluu myös yhtenä kokonaisuutena testidatan syöttäminen ja datan pitäminen validina testausta ja eri testitapauksia varten (van Veenendaal 2012, s. 74). Tietyn tyyppisissä projekteissa ympäristön laitteiston ja asetusten, viestintäliikenteen ja ohjelmistojen pitää olla samantasoinen kuin varsinaisessa tuotantoympäristössä, jotta voidaan suorittaa erilaiset testitapaukset (van Veenendaal 2012, s. 72-73). Testiympäristössä on myös hyvä huomioida erilaiset laitetyypit ja esimerkiksi vanheneva laitteisto, koska vanhaa laitteistoa on yleensä ainakin jossain edelleen käytössä. Testausympäristön kehittäminen kuuluu osana testausympäristön hallintaa.

3.3.4 Taso 3: toimintatavat

Tasolla kolme TMMi:ssä on tärkeää, että testauksen laatua kehitetään tason kaksi lähtökohtien vaatimuksien mukaisesti, kuten kehittämällä testaajien osaamista tarjoamalla testaukseen liittyvää koulutusta. Organisaation tukee testausta tällä tasolla ja organisaatio sietää riskejä, jotka saattavat vaikuttaa testaukseen suorittamiseen laadukkaasti. Testauksen elinkaari on tunnistettu tärkeäksi osaksi projektia ja sen on avainasemassa yrityksessä tai projektissa. Testaaminen on tärkeäksi koettu osa ohjelmiston kehitystyötä ja testaamisen suunnittelu aloitetaan hyvissä ajoin ennen ohjelmiston käyttöönottoa. Tarvittaessa testaus suoritetaan myös määritellyin osin ei-funktionaalisille kokonaisuuksille tai osille ohjelmistoa, jolloin testauksen avulla testataan ohjelmistoa myös muullakin tavoin kuin moduulitestauksen tai varsinaisten testitapausten avulla. Näitä testaustapoja voivat olla esimerkiksi käytettävyyteen tai luotettavuuteen liittyvät testitapaukset, jotka suoritetaan liiketoiminnan ohjauksella ja priorisoinnin avulla (van Veenendaal 2012, s. 11).

Yksi seikka, joka tasolla kolme korostuu, on vertaiskatselmuksien järjestäminen ja katselmointien tuominen. Katselmointien avulla voidaan tehdä vertailua testitulosten ja testauksen etenemisen osalta edellisten viikkojen suhteen projektin sisällä tai yrityksessä. Katselmointi ei tarkoita vain testauksen katselmointia, vaan testaajat osallistuvat aktiivisesti myös katselmuksiin, joissa ohjelmistovaatimuksia ja määritysdokumentaatiota käydään läpi. (van Veenendaal 2012, s. 11). Testausprosessin kehittämisen näkökannalta tasolla kolme kehityskohteet ja tarkastelukohteet van Veenendaalin (2012, s. 12; 79-135) mukaan ovat:

- **Testausorganisaatio**, eli sen kuvaaminen ja testaukseen käytettävät roolit ja toiminnot, urapolku, sekä testauksen organisointiin liittyvät prosessiin liittyvät parannukset.

- **Testauksen koulutusohjelma**, mikä käytännössä tarkoittaa testauksen koulutuksen järjestämistä ja sen organisoimista, mutta myös koulutuksen kehittämistä yli organisaatiorajojen.

- **Testauksen elinkaari ja integrointi**, eli käytännössä tässä kokonaisuudessa pyritään testauskäytännöt vakiinnuttaminen osaksi organisaation toimintaa ja testauksen kehityspolkujen sitominen osaksi organisaation kehitystoimia sekä testaussuunnitelman mallipohjan toteuttaminen tulevia testaustoimintoja varten.

- **Ei-toiminnallisen testaamisen** mukaan ottaminen osaksi testaamista tarkoittaa myös ei-toiminnallisten ominaisuuksien riskien määrityksen ja niiden hallinnan, sekä edelleen ei-toiminnallisten testien analysoinnin ja suunnittelun. Ei-toiminnallisten testien suorittaminen tehdään muun

testauksen ohella ja tällöin suoritetaan valitut ei-toiminnalliset testaukset sekä raportoidaan testauksien tulokset.

- **Katselmointien** tuominen mukaan ohjelmistojen testaukseen tuo vertailupohjaa eri kokonaisuuksien testauksen tilasta ja ennalta sovitut hyväksymiskäytännöt testauksen tuloksiin liittyen. Myös dokumentaation katselmointia toteutetaan osana prosessia.

Testaukseen käytetystä ajasta ja sen hyödyistä voidaan tehdä jonkinlainen laadullinen yhteenveto esimerkiksi eri projektien välillä. Näin voidaan saada tietoa, että missä painopistealueissa tai missä testauksessa on edelleen kehitettävää tai mihin osa-alueeseen tulisi panostaa voimavaroja ja edelleen kehittää testausprosessin osa-alueita. Tämä puoli on tasolle neljä pääsemiseksi ehdoton edellytys ja silloin sen on oltava osa testausprosessia. Sinänsä katselmointi on TMMi:n tasolla kolme ennen kaikkea laatua parantava tekijä, jonka avulla voidaan parantaa ohjelmiston ja dokumentaation laatua (Arovaara 2008, s. 71) ja raportoida tuloksia katselmusten perusteella asiakkaalle tai muille tahoille. Testauksella projektissa on myös tietyt virstanpylväät tasolla kolme, jotka ohjaavat testaustoimintaa (Cannegieter & van Veenendaal 2013, s. 7). Katselmukset voi aikatauluttaa myös tietyin väliajoin ohjelmistoprojektin kokonaisaikatauluun nähden, jolloin aikataulu on tältäkin osin sidottu tiettyihin tarkastuspisteisiin.

3.3.5 Taso 4: mittaaminen

Tasolla neljä avainasemassa on testauksen mittaaminen. Erilaisille kokonaisuuksille, kuten esimerkiksi laadulle ja prosessillekin voidaan asettaa erilaisia mittareita, joiden avulla testausta ja prosessia voidaan mitata. Mittaristo on koko organisaation tai projektin käytössä, jolloin tuloksien näkyvyys ja prosessin tehokkuus on kaikkien halukkaiden saatavilla (Arovaara 2008, s. 71-72). Tason neljä TMMi-prosessin kehitysalueet ja tarkastelukohteen van Veenendaalin (2012, s. 11-12; 137-162) mukaan ovat:

- **Testauksen mittaus** tarkoittaa käytännössä testauksen mittariston suunnittelua ja toteutusta, sekä käyttöönottoa ja tulosten analysointia.

- **Ohjelmiston laadun arvioiminen** tuo projektiin ja organisaatiolle sille määriteltyjen laatukriteerien priorisoinnin käyttöön testauksen laaduntarkkailussa ja lisäksi laatukriteereitä kehitetään, mitataan ja hallitaan osana testausprosessia.

- **Edistyneiden vertaiskatselmuksien suorittaminen** takaa katselmusten kehittämisen eri pisteissä määritellyn laajuuden mukaisesti ja vertailun laatukriteereitä vastaan. Lisäksi edistyneiden katselmuksien avulla voidaan kehittää testauksen laajuutta eri projekteille ja tehdä vertailua ohjelmiston koko elinkaarta silmällä pitäen.

TMMi:n tasolla neljä van Veenendaalin mukaan (2012, s. 11) testaustavat ja niihin liittyvät käytännöt ovat kokonaisuudessaan määritelty ja testaus on osa organisaation toimintatapaa ja mitattava prosessi. Mittaamisen avulla testausta ja prosessia on mahdollista kehittää ja testauksen läpiviemiselle voidaan asettaa tavoitteita. Mittaustuloksien vertailu eri projektien tai testauskokonaisuuksien kesken on mahdollista. Mittaamisen pohjalta voidaan tehdä yhteenvetoja ja parantaa prosessia ja tarkastella parannuskeinojen vaikutuksia eri kokonaisuuksien näkökannalta. Testausprosessin avulla voidaan parantaa myös tuottavuutta, sillä prosessi on osa projektin tai yrityksen toimintatapaa. Ennen kaikkea mitattavuus tuo prosessin näkökulmasta ennalta arvattavuutta ainakin aikataulutukseen ja resursointiin. Mittaustulosten avulla on mahdollisuus organisaatiossa tai projektissa asettaa laatuvaatimukset ja edelleen kehittää laadun arvioinnin metriikoita ja laatumittareita. Ennen kaikkea tällä tasolla tuotteen laatu ymmärretään tärkeänä laatutekijänä ja sen mitattavuus on ennalta määritellyn mittariston avulla mahdollista. Organisaatiossa tai projektissa voidaan myös asettaa tavoitteita, jotka koskevat koko ohjelmiston elinkaarta. Käytännössä mittaamiseen liittyvät kriteerit ja laadunhallinnan parantaminen on määriteltävä yrityskohtaisesti ylilaadun välttämiseksi.

Katselmoinnit ovat vakiintunut käytäntö organisaatiossa ja niissä todettujen testaustulosten on täytettävä ennalta asetetut tietyt laatuvaatimukset. Van Veenendaalin (2012, s. 11) mukaan tarkastuspisteet muuntuvat tuotteen laatua mittaaviksi pisteiksi, jolloin tietyissä pisteissä voidaan tavoitella aina perustason laatua korkeampaa laatua tai tavoitetason laatua. Mitattavuus on mahdollista sekä ohjelmiston elinkaaren osalta että prosessin kehitystyön osalta, koska vertailudataa on kerättynä ja jos samansuuruisten kokonaisuuksien tai samaa prosessia noudattavan ohjelmistoprojektin arviointi tehdään suhteessa olemassa olevaan historiadataan.

3.3.6 Taso 5: kehittäminen

Edellisten tasojen kehittäminen testauksen osa-alueella ja saavutukset takaavat tai näiden tulisi taata se, että organisaatiossa on mahdollisuudet toteuttaa testausta ja testaukseen liittyvät prosessin on myös kuvattu ja dokumentoitu. TMMi:n tasolla viisi on ennen kaikkea kyse prosessin parantamisesta esimerkiksi laatumittareiden avulla. Testausta kuten muitakin ohjelmistokehityksen osa-alueita kehitetään uudistamalla ja parantamalla testauskäytäntöjä ja myös hiomalla prosesseja testauksen näkökannalta. Testausprosessin parantamisen kannalta keinoja voivat esimerkiksi olla: tavoitteellinen

johtaminen, tilastointi ja tätä myöten ennustettavuus, mahdollisesti testauksen automatisointi, testitapausten tehokas uudelleen käyttö ja prossin parantaminen. (van Veenendaal 2012, s. 12).

Tällä tasolla on ominaista ennen kaikkea toimintatapojen kehittäminen ja niiden optimointi edelleen parempaan suuntaan (Arovaara 2008, s. 72). Jatkuva kehittäminen on organisaatiossa mukana prosesseissa. Erilaiset laatumittarit ja mittavuus testauksen eri vaiheissa ovat osana organisaation laatujärjestelmää. Ohjelmistotestaukseen on käytettävissä tekijöillä sopivat ohjelmistot ja muut tarvittavat työkalut testauksen toteuttamiseksi ja näitä kehitetään tarpeen mukaisesti. Tason viisi tärkeimmät osa-alueet van Veenendaalin (2012, s. 12; 164-202) mukaan ovat:

- **Virheiden ennaltaehkäisy**, mikä tarkoittaa yleisten tiedossa olevien virheiden määrittämistä ja näiden tunnistamista sekä näiden eliminointia systemaattisesti.

- **Kokonaislaadun hallinnan** avulla pyritään vakiinnuttamaan testausprosessi niin pitkälle kuin mahdollista, jolloin sen mittaaminen on mahdollista ja voidaan puuttua testauksen suorittamiseen, jos sen suorittaminen ei noudata aikaisemmin mitattua keskiarvoa. Tässä on myös tärkeää määrittää perustuotteen ominaisuudet tai eri ohjelmiston kokonaisuudet, joita kohtaan vertailudata kerätään.

- **Testausprosessin optimointi** tarkoittaa niiden testauksen kokonaisuuksien valintaa, joita halutaan parantaa ja eri prosessin osa-alueiden jatkuvaa kehittämistä. Testausprosessin parantamiseen liittyvät havainnot täytyy myös käyttöönottaa ja optimoida organisaatiolle sopiviksi. Tähän prosessialueeseen liittyy uusien testausteknologioiden arviointi ja käyttöönotto, sekä testaukseen eri projekteissa käytettyjen testien tai tapojen uudelleen hyödyntämisen arviointi.

Prosessin kehittämistä varten voidaan esimerkiksi perustaa testauksen prosessin kehittämiseen erikoistunut ryhmä, joka arvioi, tarkastelee ja tutkii prosessia tietyn väliajoin eri projektien avulla ja kehitysryhmän avulla voidaan edelleen parantaa testauksen prosesseja. Ryhmän voi aluksi toimia epämuodollisesti, mutta viimeistään tasolla viisi ryhmän olemassa olo on vaatimus. Myös testaustapojen kehittäminen ja uudelleenkäytettävyys ja näiden osa-alueiden tunnistaminen on avainasemassa testausvaiheen nopeuttamiseksi ja laadun parantamiseksi. Esimerkiksi regressiotestausta voidaan automatisoida, testitapausten dokumentointia ja hallintaa tehostaa esimerkiksi raportointia parantamalla, virheitä vähennetään analysoimalla niitä ja vähentämällä virheisiin johtaneita syitä tai yleisiä parhaita käytäntöjä tuodaan koko organisaatioin toimintatavoiksi. (van Veenendaal 2012, s. 12).

3.4 Yhteenveto

Ohjelmistoprojektissa on tärkeää valita kehitysmalli, joka tukee ohjelmistokehitystä ja on tuttu toteuttajille tai muussa tapauksessa varata tarpeeksi aikaa mallin kouluttamiseen ja siihen tutustumiseen. Ketterät menetelmät takaavat kehityksen iteratiivisesti, jolloin myös ohjelmistotuotteen ominaisuuksia voidaan testata heti niiden toteutuksen jälkeen. Siinä myös yhteinen vastuu korostuu. Prosessipohjaisessa vesiputousmallissa taas yleensä toteutetaan yksi vaihe ennen seuraavan aloittamista vaikkakin nämä voivat osittain mennä toistensa kanssa lomittain. Testausprosessien kehittäjien sekä kirjallisuuden perusteella testaaminen on hyvä aloittaa mahdollisimman ajoissa, joten tältä pohjalta myös ohjelmiston ensimmäisen version testaaminen olisi hyvä aloittaa heti, kun se on mahdollista. Kummassakin mallissa viestintä projektin sisällä, toimittajan ja asiakkaan kesken sekä sidosryhmille on avainasemassa projektin onnistumiseksi.

Altéa-projektissa oli käytössä prosessipohjaisen mallin mukainen tuotantotapa toimittajallamme, eli siihen ei voinut helposti soveltaa Scrum-mallia, jossa testauksen raportointia olisi esimerkiksi käyty läpi päivittäin. Tämä ei edes ollut mahdollista, koska testaus suoritettiin testillä ja edelleen tuotannon beta-versiolle ohjelmistoversiolla, joka oli saatavilla ennalta sovitun aikataulun mukaisesti. Näihin versioihin vietiin ne toiminnallisuudet, jotka oli alustavasti testattu Amadeuksen toimesta. Viikoittaisen edistymisen seuranta oli tarpeeksi riittävä ja toimiva tapa toimia projektissa, ja testauksen edistymistä seurattiin näin viikkoja ja kuukaudesta toiseen. Viikkoraportin malli löytyy liitteestä 6. Tiettyjen moduulien ja kokonaisuuksien testausta ja sen edistymistä seurattiin tarkasti ja testauksen painopistealueita tarkennettiin projektin edistymisen aikana. Dokumentaation tarkoituksena oli tärkeää pitää testausdokumentaatio ajan tasalla ja jokaisen testaajan tuli raportoida viikkotasolla testauksen edistyminen, jotta myös toimittajalla oli tiedossa, miten Finnairilla projektin testaus eteni. Näin tiedettiin sekä Amadeuksella että Finnairilla kuinka pitkällä testauksessa oltiin ja milloin oli mahdollista aloittaa ohjelmistojen asennukset käyttökohteisiin, testaus näissä työpisteissä ja viimein tuotannon käyttöönotto.

TMMi:n virallisen arvioinnin suorittaa vähintään kaksi henkilöä, joilla on TMMi:n Foundation sertifikaatit. Epämuodollinen arviointi tuottaa kuitenkin myös tarkan kuvan ohjelmistotestauksen ja prossien laadusta, joten Finnairin projektissa arviointi tehtiin lähinnä projektin sisäisesti. Virallisen arviointisuunnitelmaan tulee Goslinin (2009, s. 10-11) mukaan kirjata seuraavat asiat:

- arvioinnin tarkoitus
- arvioinnin laajuus
- arvioinnin käytännön toteutus, eli resurssit ja vastuut, aikataulu ja tarkennettu laajuus arvioinnin suhteen
- arvioinnin menetelmät, jotta se voidaan peilata TMMi:n teoriaan ja

- arvioinnin hyväksymiskriteerit, joita vastaan kerättyä tietoa ja dokumentteja verrataan.

Yleensä arvioinnissa käytetään hyödyksi TAMAR-kehikkoa, joka noudattaa myös osittain ISO 15504-2 -standardia ja vaatimuksia (Goslin 2009, s. 19-20; Arovaara 2008, s. 73). Varsinainen tieto arviointia varten voidaan kerätä erilaisten dokumenttien avulla, haastattelemalla tai kyselyiden avulla. Arvioinnissa voidaan käyttää arviointitaulukko kunkin prosessialueen suhteen, joiden avulla varsinainen arvosana kustakin prosessialueesta voidaan antaa. Arvosteluasteikko on viisiportainen, joista alin on ”ei arvioitu” ja korkein taso: ”täysin saavutettu”. Mukana arvioinnissa on myös vaihtoehto ”ei arvioitavissa”, jolloin kokonaisuutta tai osa-aluetta ei joko arvioida tai se ei ole arvioitavissa (Goslin 2009, s. 21).

TMMi ei sinänsä ole varsinainen prosessin kehittämiseen tarkoitettu malli, joten sen rinnalla on hyvä käyttää prosessien muutoksiin ja jatkokehittämiseen jotain toista prosessien kehittämiseen tarkoitettua mallia. Tämä tekee osittain TMMi:stä raskaan käyttää, mutta se antaa hyvän lähtökohdan erityisesti testauksen kehittämiseen organisaatioissa, jotka haluavat saada testauksen käytännöt ja prosessit kuntoon. Mallin soveltaminen vaatii aikaa, sillä muutos organisaatiossa ja mallin soveltaminen ei tapahdu hetkessä. TMMi:n mallin mukaan testauksen kehittämistä ei aina tarvitse noudattaa täysin orjallisesti. Paras testauksen prosessimalli ja testaustoimintojen hallittavuus organisaatiossa syntyy siten, että malli sovelletaan organisaation tarpeisiin sopivaksi (Goslin 2009, s. 6).

Testauksen raportoinnin mallia kehitettiin projektin etenemisen aikana. Viikoittainen raportointi ja testausprosessin kehittyminen näkyi paremmin toisessa osa-projektissa, sillä tiedossamme olivat hyvissä ajoin mahdolliset tulevat haasteet, riskit ja ennustettavuus tulevien töiden priorisoinnissa. Tämän lisäksi raportointi kertoi viikoittain, miten projektissa edettiin ja miten etenemisemme vaikuttaisi aikatauluun jonkin osa-alueen osalta, tai millä tavoin meidän tulisi panostaa tiettyihin projektin tai testauksen kokonaisuuksiin. Testauksen prosessin kehittäminen oli avainasemassa eri kokonaisuuksien hallinnoimiseksi ja testauksen eri kokonaisuuksien parantamiseksi.

4 TESTAAMINEN PROJEKTISSA, TESTAUKSEN LÄPIVIEMINEN JA TESTAUSTAVAT

Keskityn tässä työssä tarkemmin testausprosessin kuvaamiseen, eli miten testauksen kanssa toimimme Finnairilla Altéa-projektissa ja kuinka testaukseen liittyvät asiat kehittyivät Finnairilla projektin aikana. Testauksen perusteisiin kuuluvat erilaiset projektille valitut testaustavat, dokumentointi, raportointi, virheiden hallinta, testausstrategia ja pitkällä aikavälillä myös testauksen ja testausprosessin kehittäminen. Tässä ja seuraavassa luvussa avaan näitä asioita Finnairin projektin näkökulmasta liittyen testauksen prosessin kehittämiseen ja raportointiin. Tässä on hyvät lähtökohdat, joita voi ottaa huomioon testausta suunniteltaessa ja testaustapoja organisaatioon sovellettaessa. Testausprosessin esittely projektin aikana ja projektille tyypillisimmät testaustavat ja raportointi on esitelty tarkemmin, kuten projektin aikana toimittiin. Nämä on pyritty kuvaamaan tässä ja seuraavassa luvussa mahdollisimman kattavasti.

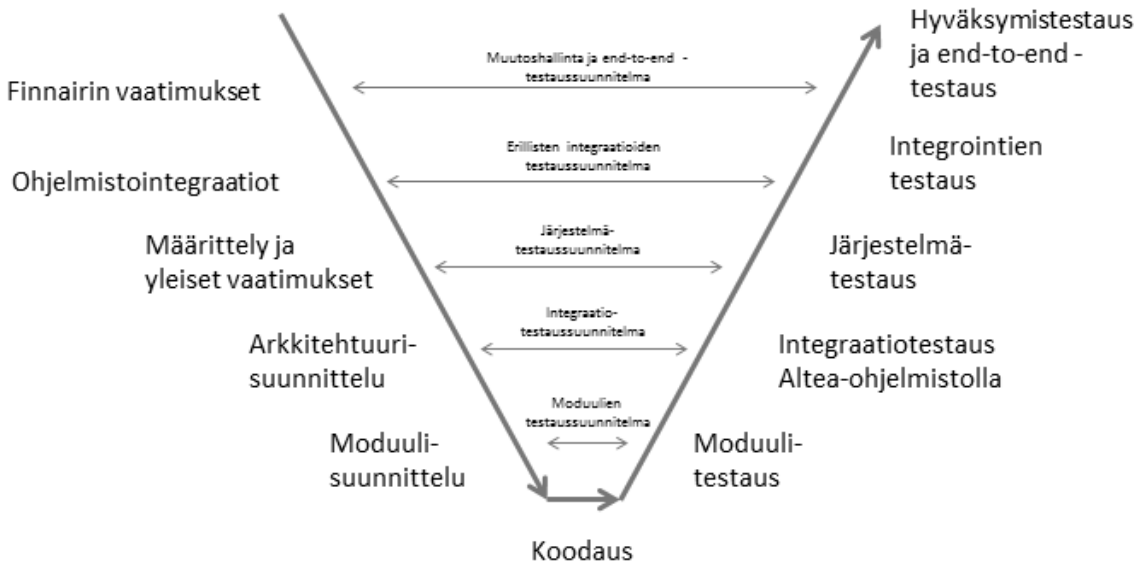
4.1 Testausmalli

Testausmalli muistutti pitkälti Altéa-projektissa testauksen V-mallia (Haikala & Märijärvi 2002, s. 287), missä toteutettiin myös peruskoodin ja ohjelmiston perustestaaminen, moduulitestaaminen moduuleittain, moduulien ja ohjelmistointegraation testaus ja merkittävämpänä testauksen muotona järjestelmätestaus. Lähtökohtana oli pitkälti toteuttaa testaus toiminnallisen määrittelyn perusteella ja peilaten ohjelmiston toimintaa loogisen ajattelun mukaisesti toimivaksi, jonka perusteella voidaan olettaa ohjelman toimivan tietyllä tavalla. Lisäksi mukana oli ohjelmistoon syötteiden avulla tuotavien ohjelmistojen ja mahdollisten ulkoisten ohjelmistojen integraatioiden tai syötteiden testaaminen systeemitestauksessa. Kokonaisen järjestelmän järjestelmätestaus tai Altéa-projektissa end-to-end -testauksen yhteydessä tehtiin koko ohjelmiston kattava testaus, missä käytännössä tiettyjen testitapauksien avulla toteutettiin ohjelmiston koko toiminnallinen elinkaari ensimmäisestä toiminnosta päätepisteeseen saakka yhden lennon osalta.

Testauksen apuna projektin aikana oli ohjelmistoista toteutetut määritysdokumentit ja UML-kuvaukset ja niiden pohjalta toteutetut käyttötapauskaaviot osasta toiminnallisuuksia. Näiden ja oman alakohtaisen osaamisen avulla pystyimme tuottamaan tarkemmat testitapauksen ja niiden variaatiot. Lähestymistapa testaukseen oli lähinnä mustalaatikkotestausta (black-box-testaus), mutta osittain seassa oli toisinaan

myös harmaalaatikkotestaamiseen viittaavaa lähestymistapaa. Mustalaatikkotestaamisessa testitapaukset toteutetaan määritystä vastaan (Haikala & Märijärvi 2002, s. 289). Nämä testitapaukset myös prosessin mukaisesti vielä tarkastettiin toisella järjestelmäasiantuntijalla, jotta niihin oli mahdollista tehdä lisäyksiä, kehittää testitapauksia tai varmistaa, että kaikki tulisi testattua.

Testauksen V-malli koostuu laajennettuna seuraavista osa-alueista (Haikala & Märijärvi 2002, s. 287), jotka on esitelty kuvassa 4.1.



Kuva 4.1. *Testauksen V-malli Finnairin projektissa. Seuraavassa kappaleessa avattu V-mallin käyttö Finnairin projektissa.*

Finnairilla testauksen kokonaisuus oli jaettu seuraavalla tavalla (suluissa testauksen vastuutahoja):

- Yksikkötestaus tai ts. moduulitestaus (Amadeus ja ITC Infotech).
- Integrointitestaus (Amadeus, Qantas, Finnair sekä ITC Infotech).
- Järjestelmätestaus: ohjelmiston toimivuus (Finnair ja Qantas).
- Ohjelmistointegraatioiden testaaminen, eli systeemitestaus. Testattava, verifioitava ja validoitava eri järjestelmissä tuotetun datan siirtyminen kustakin järjestelmästä Altéa FM -ohjelmistoon (Finnair ja sovelluksien omistajatahot, kuten rahti, reittiliikenne ja muut tahot).
- Muutoshallinta: testaus muutostöiden ja Finnairin vaatimusten mukaisesti (Finnair ja ITC Infotech).
- End-to-end -testaus, joka vastaa lopullista hyväksymistestausta kokonaisjärjestelmän toiminnallisuuksien osalta (Amadeus ja Finnair).

Amadeus oli toteuttanut ohjelmiston omat testit ohjelmistokehityksen eri tasoilla, joiden perusteella ohjelmisto voitiin tuoda asiakkaiden testattavaksi ja toimitettavaksi. Altéa-käyttöönottoprojektin osalta täytyy myös huomioida, että Finnair oli toinen lentoyhtiö, jossa Altéa FM -sovellus otettiin käyttöön ja sen takia sovellusta voitiin pitää vielä joltain osin keskeneräisenä järjestelmänä. Qantaksen ja Finnairin tehtävänä oli osittain auttaa toimittajaa mahdollisten puutteellisten ominaisuuksien määrittelemiseksi sekä ohjelmiston kehittämiseksi. Tällä oli omat vaikutuksensa projektin läpivientiin ja erikseen myös kustannuksiin.

4.2 Testauksen strategia

Lähtökohtana testausstrategialle oli testata ohjelmisto sekä tuotetasolla että toiminnallisuuksiltaan käyttötarkoitusta vastaavaksi. Sen lisäksi tiimimme vastuulla oli ohjelmistojen koulutus ja asennukset eri toimipaikkoihin Amadeuksen kanssa yhteistyössä. Koko projektin tavoite oli saavuttaa hyväksytty ja testattu ohjelmisto, jonka käyttöönotto oli mahdollista ilman tunnistettuja riskejä. Edellytykset vaativat myös kaikkien ohjelmistoa käyttävien koulutuksen suunnittelun ja koulutuksien toteuttamisen. Projektissa strategisesti merkittävässä osassa olivat myös käyttöönoton suunnittelu ja varsinainen käyttöönotto ja käyttöönoton tuki.

Jokaiselle järjestelmäasiantuntijalle asetettiin henkilökohtaiset tavoitteet ja näiden saavuttamista seurattiin. Tämä tapahtui asettamalla tietylle henkilölle esimerkiksi tietyt testiosa-alueet tai vastaamaan omasta vastuunmukaisesta erillisestä kokonaisuudestaan. Tämä henkilökohtainen projektissa asetettu tavoite oli myös sidottu yhtiössä osittain henkilökohtaiseen palkkausosaan. Projektin kokonaistavoite oli saada Finnairin laivasto käyttämään uutta sovellusta lentojen tasapainolaskelmien tuottamisessa ja prosessimuutokset tähän kokonaistoimintaan tietyn ajan sisällä. Tässä ei voitu kuitenkaan ottaa minkäänlaisia riskejä, joten aikataulutuksen muutokset olivat selkeästi suotuisampia, kuin lähteä Altéa FM -sovelluksella ohjelmiston käyttöönottoon, jonka jälkeen järjestelmässä olisi huomattu mahdollisesti vakavia puutteita.

Priorisointi projektin etenemisen kannalta suoritettiin sen mukaisesti, miten kriittisesti tietyt moduulit ja näiden toiminnalliset osa-alueet tulisivat mukaan ohjelmiston käyttöönotossa Osittain joidenkin moduulien käyttöönotto ja muutoksen aiheuttama lopullinen paine oli muutosvastarinnan osalta sen verran iso, että kaikkea ei pyritty muuttamaan kerralla ohjelmiston käyttöönoton yhteydessä. Testauksen hallinnan avulla oli tärkeää priorisoida eri moduulien testaus tulevan käytön mukaisesti, eli sen mukaan, kuinka tärkeää kunkin moduulin käyttö oli lentokoneen tasapainolaskelman tuottamisen kokonaisprosessin mukaisesti ja käyttöönoton onnistumisen kannalta. Kriittisiä kokonaisuuksia olivat lennonvalmistelijan moduuli, johon Altéa FM -järjestelmästä kaikki tiedot ohjelmiston moduuleista kootaan, ja matkustajatietojen hallintamoduuli sekä näiden tietojen siirtymisen varmistaminen Altéa FM -järjestelmään. Tasapainolaskelman toteuttamisen päämoduuli, johon kaikki

järjestelmään syötetyt tiedot kootaan, on Altéa FM -ohjelmistossa avainasemassa. Myös viestiliikenteen siirtymisen varmistaminen järjestelmien välillä oli tärkeässä asemassa testauksessa, jotta saatiin varmistettua, että tieto siirtyi eri järjestelmien välillä. Viestiliikenteen estyessä järjestelmien välillä voitiin tarvittaessa toimia varajärjestelyin. Tulevan tuotantokäytön tärkeimpien moduulien testaus ja toimintojen varmistaminen toi projektiin tiettyjen osa-alueiden priorisointia suhteessa muihin ohjelmiston moduulien korjausten priorisointiin.

Vähemmän kriittisiä moduuleita käyttöönotossa olivat lentopetrolin syöttämisen mahdollisuus tai rahtimoduuli, joiden käyttö voitiin ohittaa esimerkiksi tarjoamalla sama tieto viestin avulla telexillä tarvittaville tahoille. Valitettavaa oli myös se, että osa sovelluksen ominaisuuksista ei täysin täyttänyt niitä kriteereitä tai määrityksiä, joiden perusteella toiminnallisuudet olisi pitänyt toteuttaa. Tämä toi projektin priorisointiin moduulien kokonaisuuden hyväksymiseen omat lisähaasteensa.

Projektin johto määritteli riskit, joista selvimmät olivat aikatauluhaasteet ja ulkoisten ohjelmistojen integraatioiden ja viestiliikenteen toimivuus. Jos esimerkiksi integraatiota Finnairin AY FIDO -lähtöselvitysjärjestelmästä ei saada toteutettua aikataulun mukaisesti, niin Altéa FM -ohjelmiston käyttöönottoa ei voitaisi tehdä. Ohjelmistojen tulevat tuotantopäivitykset ja ohjelmiston käyttäjähallinta asettivat myös haasteita Altéa FM -ohjelmiston käyttöönoton osalta. Myös testauksen edetessä ilmeni, ettei aivan kaikki sovelluksen osalta ollut kaikki valmista, kuten toimittaja antoi ymmärtää. Tämä toisaalta toi lisää aikataulupaineita projektin ja testauksen läpiviemiselle. Esimerkiksi kuormausmoduuliin oli jäänyt toteuttamatta kaikkien toiminnallisuuksien käyttäminen näppäinkomennoin. Altéa FM -projekti venyikin sille asetetusta alkuperäisestä aikataulusta, koska ohjelmistoon pyydetyt muutokset Finnairille eivät olleet aikataulun puitteissa valmiita ja niitä ei pystytty testaamaan ajoissa. Myös se seikka, etteivät ohjelmiston toiminnallisuudet täyttäneet määrityksessä mainittuja toiminnallisuuksia, oli selkeä este ohjelmiston käyttöönotolle alkuperäisessä aikataulussa. Tärkeintä oli kuitenkin tämän nojalla poistaa riskitekijät ohjelmiston käyttöönoton yhteydessä, mikä olisi mahdollisesti jollakin tavalla vaarantanut ilmailusäädöksiä tai pahimmassa tapauksessa ihmishenkiä. Projektin aikana oli turvallisempaa ensisijaisesti siirtää käyttöönottoa selkeiden välitavoitteiden avulla, kuin ottaa joltain kriittisiltä ominaisuuksiltaan puutteellinen ohjelmisto käyttöön.

4.3 Testauksen suunnittelu

”Testaus testaa järjestelmän toiminnallisia ja ei-toiminnallisia vaatimuksia, eli mittaa järjestelmää testitapauksien avulla”, toteaa Stenberg (2007a, s. 7). Sommerville (2007, s. 516) esittää, että verifiointi vastaa kysymykseen: ”Toteutammeko tuotteen sellaiseksi kuin sen määrittelimme?”. Verifioinnin avulla on määrä selvittää ohjelman toiminnan oikeellisuus, verraten tuotetta esimerkiksi sen spesifikaatioon. Verifiointi toteutetaan yleensä ohjelmistoprojekteissa sovelluksen testauksena. Näistä lähtökohdista lähti myös

Altéa-ohjelmistotuotteiden testaaminen ja projektin perustarkoitus oli saada tuote hyväksyttyä projektin aikana hyväksymistestattua Finnairin käyttöön ja käyttötarkoituksia vastaavaksi. Testitapausten suunnitteluun ja testitulosten dokumentoitiin käytettiin projektissa parhaaksi havaittua tapaa, eli Microsoft Excel-taulukkotiedostoa. Tästä löytyy mallipohja liitteenä 5. Tähän päädyttiin siksi, koska testaukseen kehitettyjen sovellusten tutkimiseen ei ollut projektin aikana aikaa ja MS Excel oli helposti kaikkien saatavilla ja dokumentaatioon olisi mahdollista myös myöhemmin tarvittaessa tehdä muutoksia sekä tutkia testituloksia ilman, että siihen tarvittaisiin tietynlaista ohjelmistoa ja mahdollisia erillisiä lisenssihankintoja. Jokaisen moduulin osalta MS Excel -tiedostossa oli tietty rakenne, missä oli ensimmäisellä välilehdellä sisällysluettelo, seuraavalla sivulla muutoshallinta ja seuraavalla välilehdellä tarkastajan kommentointi. Neljännellä välilehdellä oli varsinaiset testitapaukset erilaisin mahdollisin skenaarioin, jotka oli toteutettu käyttötapauksiin perustuen, sekä Finnairin tarpeisiin Altéa FM -ohjelmiston toiminnallisuuksien osalta. Hetzelin (1997, s. 128) mukaan testitapauksien suunnitteluun liittyvät seuraavat asiat:

- testauskokonaisuuksien ryhmittely omiin kokonaisuuksiinsa
- testaustavoitteet
- testitapauksen varsinainen kuvaus
- testin suorittamiseen vaadittava data tai testin esivalmistelu
- odotettu lopputulos
- testin esivalmistelut tai käyttötapaus, jonka jälkeen testaus voidaan aloittaa
- testin suorituksen kuvaus testauksen jälkeen
- hyväksymiskriteerit, eli oletettu lopputulos
- testauksen suorittaja tai vastuuhenkilö
- suoritusloki testin osalta ja
- testin lopputulos, eli mihin tulokseen testissä päädyttiin.

Projektissa MS Excel -tiedostoihin raportoitiin testitapausten yhteyteen testauksen eteneminen, mahdollinen testauksen lopputulos ja ne seikat, jotka mahdollisesti estivät esimerkiksi jonkin testitapauksen edistämisen. Näitä olivat esimerkiksi mahdolliset virheet ja näiden virhenumero sekä kuvaus, virheen tilanne tai muutoshallinnan vaikutus testitapaukseen. Katso malli tästä dokumentista liitteestä 5, jossa on malli käytössämme olleesta testitapausten dokumentoinnista taulukkomuodossa. Dokumentista voi hyvin havaita, että Hetzelin esittämät testitapausten suunnitteluun liittyvät asiat löytyvät melko pitkälti myös Altéa-projektin testitapausten dokumentaatioista. Dokumentista oli helppo hakea myös vertailutiedot testauksen viikkoraporttiin (malli tästä on liitteessä 6: viikkoraportoinnin malli Altéa-projektissa), jonka avulla testauksen etenemisen raportointi käytiin viikoittain läpi. Stenbergin mukaan testauksessa voi hyödyntää esimerkiksi seuraavia lähestymistapoja testaamiseen (Stenberg 2007a, s. 9):

- Toimivatko järjestelmät (toiminnalliset vaatimukset), mikä tarkoittaa ohjelmiston eri toiminnallisuuksien ja näiden testaamista mahdollisimman kattavasti alatasolta esimerkiksi eri moduuleissa ja ohjelmiston määritellyn laajuuden mukaisesti ohjelmiston ylätasolle saakka.

- Miten se toimii (ei-toiminnalliset vaatimukset), kuten esimerkiksi koko järjestelmän testaaminen tiettyjen ennalta määriteltyjen lähtökohtien mukaisesti tai järjestelmän rakenteen ja arkkitehtuurin tarkastaminen. Muita ei-toiminnallisen testauksen muotoja ovat esimerkiksi luotettavuustestaus, kuormitustestaus, rasitustestaus, käytettävyystestaus tai ylläpidettävyyden testaaminen (Kollanus 2006, s. 2).

Näistä toiminnallisten vaatimusten testaaminen liiketoimintaosaamiseen liittyen oli Finnairin testaajien vastuulla ja perustoiminnallisuuksien testaaminen yhteystyössä meidän kanssa testausta tekevän intialaisen alihankkijan vastuulla. Näin merkittävä prosesseihin ja liiketoimintaosaamiseen liittyvä testaaminen ja sen osaaminen pysyi yrityksen sisällä ja myös sen jatkohyödyntäminen tulevien päivitysten tai regressiotestauksen osalta oli helpompaa, koska ammattitaitoinen ja perehtynyt henkilöstö löytyi yrityksen sisältä. Tarvittaessa testaaminen myös näiden osa-alueiden osalta olisi mahdollista siirtää ulkoiselle taholle, jos näin jossain vaiheessa päätetään. Sen sijaan tämä perustestaaminen suoritettiin Finnairin ulkopuolella, koska sen osaamiseen hallintaan ei ollut mielekästä käyttää yrityksen prosesseja ja yrityksen sisäistä toimintaa ymmärtäviä järjestelmäasiantuntijoita. Testauksen ohella projektissa kehitettiin myös prosessia, joka liittyi koko lähtöselvitykseen, mutta myös lennon valmisteluun ja tasapainolaskelman tuottamiseen liittyviin työtehtäviin. Näin uuden ohjelmiston käyttöönoton varjolla Finnairilla saatiin myös lähtöselvitykseen liittyviä prosesseja uudistettua ja tehostettua.

Lähestymistapoja testaamiseen ovat Stenbergin (2007a, s. 11) mukaan kaksi erilaista tapaa toimia:

- positiivinen testaaminen, joka pyrkii osoittamaan, että järjestelmä toimii, niin kuin on suunniteltu ja sovittu ja
- negatiivinen testaus, jossa pyritään löytämään mahdollisimman paljon virheitä, koska järjestelmä ei toimi niin kuin se on suunniteltu.

Altéa-projektissa testaamistarkoitus oli negatiivinen lähestymistapa, jolloin pyrimme todentamaan, ettei järjestelmä toimi niin kuin on määrityksen mukaisesti sen suunniteltu toimivan ja pyrimme osoittamaan siinä ilmenevät virheet. Näin testaajat pystyivät myös tuomaan järjestelmään liittyvät kehitystoimet ilmi. Usein tilanne oli se, että järjestelmä toimi kuten se oli määritelty, mutta se ei vain toiminut oikein Finnairin käyttötapauksiin tai -tarkoituksiin, eli se ei tukenut Finnairin toimintatapoja. Joitakin ohjelmistojen

toiminnallisuuksia jouduttiin tarkentamaan muutoshallinnan kautta ja myös jo ennalta määriteltyjen toiminnallisuuksien toteuttamiseen tarvittiin korjauksia ohjelmistokehityksen avulla.

Testauksen varsinainen suunnittelu perustui ohjelmistomääritysten läpikäymiseen. Tärkeimmäksi asiaksi tässä projektissa osoittautui testitapauksia suunniteltaessa ja toteuttaessa järjestelmäasiantuntijoiden substanssi- ja liiketoimintaosaaminen. Järjestelmäasiantuntijoiden aikaisempi osaaminen käytännön töistä lentoyhtiössä oli merkittävä etu testitapauksia suunniteltaessa ja toteuttaessa. Tätä osaamista myös painotettiin projektin aikana, vaikka Finnairilla olikin käytettävissä myös ohjelmiston määritykset. Järjestelmäasiantuntijoiden vuosien aikana keräämät kokemukset työtehtävien hoitamisessa, Finnairilla määritellyt toimintaprosessit eri osa-alueiden osalta määrittelivät pitkälti sen tarpeen, johon Altéa-ohjelmiston oli myös vastattava. Näin myös suuri osa käyttötapauksista ja testitapauksista suunniteltiin ja toteutettiin tältä pohjalta. Oli oikeastaan tärkeämpää toimia eurooppalaisen lentoyhtiön ja Finnairin ajatusmaailmassa ohjelman toiminnallisuuksia käytettäessä, sillä ensimmäinen käyttöönotto oli tehty ohjelmiston osalta Qantaksella. Qantaksella oli kuitenkin aivan erilaiset vaatimukset ja käyttötapaukset eri ohjelmiston toiminnallisuuksille kuin Finnairilla.

4.4 Testauksen kohteet ja kattavuus

Testauksen lähtökohdat olivat testata kaikki ohjelmiston yksiköt (rahti, lentopetroli, matkustajatietojen syöttö ja muut moduulit) ja koko ohjelmisto. Tässä yhteydessä testattiin myös Finnairin tarvitsemien ohjelmistojen mahdolliset integraatiot tai viestiliikenteellä hoidettavien tietojen syöttö verraten näitä lentoliikenteen standardeihin. Testauksen tavoitteena oli löytää Altéa FM -sovelluksesta järjestelmäasiantuntijoiden liiketoimintaosaamiseen perustuen ne poikkeavuudet, jotka olivat Finnairille tai eurooppalaisille lentoyhtiöille avainasemassa ja joita ei käyttötapauksien mukaan ollut aikaisemmin Amadeuksessa huomioitu ja toteutettuna Altéa FM -sovellukseen. Testattavana olivat myös tietyltä osin ei-toiminnalliset ominaisuudet; kuten tuotteen asennukseen liittyvät testaukset ja ohjelmistoversioiden päivittäminen sekä asennukset erilaisiin ympäristöihin. Intialaisen tiimin vastuulla oli ennen kaikkea ohjelmiston perustoiminnallisuuksien testaaminen, kuten pääperussyötteiden käyttö sovelluksessa (esimerkiksi skandinaaviset merkit ja muut vieraat muut kirjaimet), käyttöliittymä ja ohjelmiston toimivuus näppäimistön avulla. Aivan lopuksi suoritettiin vielä koko järjestelmän kattava testaaminen (end-to-end -testaus, katso kohta 4.7.3.), missä käytännössä koko lennon elinkaari käytiin läpi sen siirtymisestä Altéa FM -sovellukseen ja viimein lennon arkistoinnin onnistuminen. Tätä voidaan pitää eräänlaisena hyväksymistestauksena, mutta osa moduulien toiminnallisuuksista testattiin erikseen vielä hyväksymistestauksen osalta, jotta niille spesifiset toiminnallisuudet olivat varmasti kunnossa tarvittavilta osin. Altéa FM:n

testaamisessa kompleksisuutta läpivientiin toi se seikka, että sen sisartuote eli Altéa CM oli tulossa markkinoille vasta vuoden 2008 lopulla ja tämän välimenoajan Altéa FM:n osalta matkustajatietojen siirto piti hoitaa sinne Finnairin vanhasta AY FIDO -järjestelmästä. Tämä vaati projektin osalta merkittävää lisätyötä Finnairin, vanhan järjestelmän toimittajan ja Amadeuksen osalta. Qantas ei omassa projektissaan Altéa FM -sovelluksen osalta siirtänyt matkustajatietoja lainkaan Altéa FM -ohjelmistoon käyttöönottoprojektissaan, vaan siirsivät ne sinne manuaalisesti.

4.5 Testaustekniikat liittyen perustestaamiseen testitapausten avulla

Ohjelmistolla suoritettiin sitä varten toteutetut ja suunnitellut testitapaukset, joita oli eri moduuleihin, eli yksiköihin, suunniteltu ja dokumentoitu 30 - 80 kappaletta moduulia kohden. Parhaimmillaan yksittäisiä testauspolkuja oli merkittävästi enemmän, koska osasta testitapauksista oli monta eri variaatiota ja sen lisäksi esimerkiksi ne olivat testattava eri syötemäärällä. Samoja testejä oli myös toistettava useilla viikoilla ja eri tapauksiin liittyen useamman kerran. Näitä olivat esimerkiksi jatkolentojen tai erilaisten lentotyyppien testaamiseen liittyvät testitapaukset. Kohdassa 4.6. testitapaukset moduulien testauksessa ja sen alakohdissa esitellään esimerkein testitapauksia, jotka ovat eri moduuleihin suunniteltu.

Sommervillen (2007, s. 539) mukaan testauksessa pitäisi vähintään ottaa huomioon seuraavat kokonaisuudet:

1. Toiminnallisuudet, joihin on mahdollista päästä eri menuista, eli kaikki näytöt ja näiden toiminnallisuudet. Huomioitavaa on nykyisin myös se, että hiiren oikean näppäimen takaa löytyy lisää valikoita tai niihin on mahdollista myös päästä erilaisilla näppäinkomennoilla. Nämä näppäinkomennot olivat avainasemassa, sillä usein lentoasemilla ei ole käytössä lainkaan hiirtä käytettävissä.

2. Erilaiset kombinaatiot näiden ja muiden toiminnallisuuksien osalta, joihin vaaditaan edellisiä näyttöjä ja niiden syötteitä.

3. Tietojen syöttö, mikä on testattava oikeilla ja väärillä arvoilla.

Nämä olivat hyviä suuntaviivoja testitapausten suunnittelussa. Peruslähtökohtana oli testata kaikki kunkin moduulin toiminnallisuudet määrittelyä vastaan, mutta myös niin, että moduulien käyttäminen tuki Finnairin toimintatapaa käytettävien toiminnallisuuksien osalta. Lisäksi testattiin kaikki toiminnallisuudet ja näytöt erilaisten syötteiden avulla sekä niille siirtyminen ohjelmiston muista moduuleista.

Integraatiotestauksessa toteutettiin testitapauksia niin, että eri moduulien välinen interaktio ja toiminnallisuudet tulivat testattua ja näiden osalta tiedon siirtyminen eri kokonaisuuksien välillä oli mahdollista verifioida. Näin voitiin varmistaa, että moduulien välinen tiedonsiirto ja toiminnallisuudet toimivat eri moduulien ja liittymien välillä, kuten Sommerville (2002, s. 541) integrointitestauksen tavoitteen määrittelee. Hetzelin (1998, s. 130) mukaan integrointitestauksessa tavoitteena on saavuttaa toimivan ohjelmistorungon eri moduulien välinen transaktio ja näiden toimiva kommunikaatio. Integraatiotestauksessa suoritimme testauksen kokoavasti. eli Bottom-Up -mallin mukaisesti, mikä Haikalan & Märijärven (2002, s. 288) tarkoittaa siirtymistä alimman tason yksiköistä tai moduuleista ylöspäin. Tämä näkyi esimerkiksi siten, että testasimme eri moduulien arvojen syöttämistä ja näiden arvojen muuttamista. Tämän jälkeen tarkastelimme näiden arvojen vaikutuksia lennon tasapainolaskelmaan liittyviin arvoihin ja muutoksien vaikutusta koontinäytöltä. Lisäksi toinen hyvä esimerkki on erilaisten mahdollisten konfliktitilanteiden testaaminen, eli miten esimerkiksi liian vähäinen kuormauksen määrä tai kuormauksen väärin asettelu vaikuttuvat eri arvoihin. Testasimme myös useaan otteeseen tietojen syöttöä samanaikaisesti eri moduuleista ja tarkastelimme tietojen päivittymiseen liittyviä konflikteja ja näiden ilmoitusten käsittely eri moduuleissa.

Ei-toiminnallisuuden testauksen eräät merkittävimmät kokonaisuudet, eli rasitus- tai kuormitustestaus rajattiin projektissa testauksen osalta kokonaan testauskokonaisuuden ulkopuolelle. Näiden testaamista ei nähty meidän tapauksessa tarpeelliseksi, sillä Finnairin sopimus takasi tietyn tyyppiset hyvitykset ohjelman ollessa käyttökelvottomassa tilassa, jos syy johtuisi toimittajasta. Amadeus oli myös toimittanut ohjeistuksen ja tietoliikenteen vähimmäisvaatimuksista, joiden tuli täyttyä, jotta ohjelmistoa pystyi käyttämään sujuvasti.

Ohjelmistosovelluksen testitapaukset perustestaukseen

Nämä olivat Altéa-projektissa intialaisen ITC Infcotechin vastuulla ja suoritettiin projektiin osalta sovitulla kuormituksella ja ohjelmistoautomatisoinnilla. Periaatteessa kyseessä oli perustestausta, kuten esimerkiksi kenttien täyttämiseen liittyvät arvot ja täyttämiseen liittyvät rajoitteet, näppäinkomentojen jatkotestaaminen, ohjelmistojen valikoiden testaaminen, graafisen käyttöliittymän testaaminen eri resoluutioilla ja laitteistolla, asennustestaus eri ympäristöihin ja niiden toimivuus. Nämä testitulokset raportoitiin projektissamme intialaisen koordinaattorin toimesta testauksen etenemisen raportoinnin yhteydessä. Seuraavassa on esitelty perustestaukseen liittyviä testitapauksia, joita erityisesti intialainen alihankkija toteutti puolestamme osana projektia.

Testitapauksen yleiskuvaus laajennuksineen, esimerkki 1

Kenttien verifiointi (esimerkiksi nimi-kentät, lennonnumerot ja rahtitiedot) tehtiin syöttämällä eri kenttiin kaikki aakkoset a-å/A-Å, numerot ja näiden kombinaatiot: 0-9. Myös erilaiset erikoismerkit syötettiin eri kenttiin määrityksen mukaisesti tai erikoistarpeet huomioiden ja näiden syöttäminen kenttiin tarkastettiin. Lisäksi toteutettiin kenttien validoiminen siten, että niihin pystyi syöttämään oikean määrän merkkejä kenttää kohden, eikä tätä ole jollain tavalla estetty.

Testitulokset ja havainnot tiivistettynä

Sellainen seikka ilmeni testauksen perusteella, etteivät skandinaaviset merkit siirtyneet oikein eri järjestelmistä AY FIDO Altéa FM -ohjelmistoon. Esimerkiksi ä/Ä-kirjain siirtyi jossain tapauksissa väärin ae/AE-merkkirivinä. Ohjelmistosta puuttui tuki muun muassa yleisimmälle suomenkielessä käytetylle merkille, joka korjattiin tietyllä Javan laajennuksella. Toinen huomio oli, ettei rahtitietoihin pystynyt syöttämään kaikkia tarvittavia tietoja, koska kenttä oli yhden arvon liian lyhyt.

Testitapauksen yleiskuvaus laajennuksineen, esimerkki 2

Komentojen toimivuus ja navigointi eri moduulien välillä toteutettiin sekä hiiren että näppäimistön avulla. Myös Windows-perusnäppäimistöjen toiminnallisuudet testattiin kussakin näytössä erikseen (esimerkiksi Tab, Win+Tab, Enter, Esc). Kaikissa näytöissä testattiin myös näppäinkomennot ja mahdolliset oikopolut erilaisiin näyttöihin ja paluu näistä takaisin.

Testitulokset ja havainnot tiivistettynä

Testien perusteella havaittiin, etteivät navigaatio tai valikot toimineet samoin näppäimistökomennoin kuin hiiren avulla. Jotkin Windows-käytön standardikomennot eri kirjainyhdistelmillä eivät toimineet määrityksen mukaisesti tai ristiriidassa Windows-komentoja vastaan. Joissakin moduuleissa näppäinkomennot eivät toimineet joko lainkaan, tai sitten niissä oli puutteita.

4.6 Testitapaukset Altéa FM -ohjelmiston moduulien testauksessa ja ohjelmistojen integraatioiden testaus

Testitapaukset kirjoitettiin sen mukaisesti, kun ne oli Finnairin vaatimuksissa aikaisemmalla ohjelmistolla ja koneen käännön sekä lähtöselvitysprosessin osa-alueissa vaadittu käytännössä toimiviksi. Stenbergin (2007a, s. 18) mukaan tämä tapa, joka vastaa todellisuudessa järjestelmän käyttö on yleisin testausnäkökulma. Testitapauksiin

ja ohjelmiston toiminnallisiin vaatimuksiin vaikutti pitkälti Finnairin ohjeistus ja STM, eli Station Manuals, missä on kuvattu kaikki eri kohdealueilla tapahtuvat toiminnallisuudet ja prosessit. Kaikki lentokoneen kääntöön esimerkiksi liittyvät toiminnallisuudet ja prosessi on tarkalleen määritelty ja säännelty sekä näihin liittyvä päätöksenteko ohjeistettu. Testauksessa toteutui erilaisten testitapausten ja erityisesti käyttötapausten testaaminen, mikä on Stenbergin (2007a, s. 18) mukaan tärkeä huomioida testitapauksia suunniteltaessa. Tosin myös koko lähtöselvitysprosessin uusimisessa tehtiin töitä sen eteen, että uuteen järjestelmään ei prosessien tehostamisen takia tarvitse tehdä joka osa-alueella ylimääräisiä muutoksia aikaa vievän ja lisäkustannuksia tuovan muutoshallinnan kautta. Varsinaiset testitapaukset perustuivat dokumentoituun prosessiin, kokemukseen ja historiatietoon, mutta niiden osalta otettiin myös huomioon ohjelmiston määritysdokumentaatio. Näistä lähtökohdista toteutettiin kullekin erilaiselle käyttötapaukselle testitapaus ja erilaiset tilakaaviot dokumentoitiin MS Excelissä numeroin, jotta eri käyttötapaukset oli mahdollista suorittaa erilaisten variaatioiden osalta (katso liite 5: malli testaustaulukosta, johon testitapaukset dokumentoitiin kunkin ohjelmistomoduulin osalta ja mihin myös testitulokset kirjattiin). Käytössä testauksessa oli aitoa dataa oikeista käyttötapauksista, eli suoraan esimerkiksi olemassa olevilta lennoilta matkustaja- ja laukkutietojen tiedot, rahdista rahtitietojen oikeita tietoja ja muista osa-alueista tarvittavat tiedot. Testaamisessa hyödynnettiin suoraan eri järjestelmien autenttista dataa ja myös vertailudata vanhan järjestelmän laskukaavojen mukaisesti oli saatavilla tarvittaessa vertailudataksi. Tämä data kerättiin olemassa olevien lentotietojen perusteella ja lähtökohtaisesti tarkoitus oli kerätä erilaisia lentoja, joiden osalta voitiin suunnitella testattavaksi erilaisia konetyyppejä ja esimerkiksi näiden eri standardisoituja kuormaustapoja erilaisilla matkustajamäärillä.

4.6.1 Esimerkkejä eri ohjelmistomoduulien testauksen osalta

Seuraavissa alakohdissa on esitelty testitapauksia, jotka suunniteltiin ohjelmiston eri moduulien toiminnallisuuksien testaamista varten. Moduulien testauksen osalta oli tärkeää, että testitapaukset testattiin manuaalisen toimenpiteiden osalta, eli käyttäessä suoraan Altéa FM -järjestelmää. Testaus tehtiin myös kaikkien ohjelmistojen järjestelmäintegraatioiden osalta, jos sellainen oli suunniteltu toteutettavaksi. Tiedon lähettäminen tehtiin näin mahdolliseksi erillisestä järjestelmästä, jossa tarvittava tieto oli jo valmiina. Tällöin tietoa ei ole tarvetta erikseen syöttää käsin Altéa FM -sovellukseen kuin poikkeustapauksissa. Testitapaukset oli rajattava ja suunniteltava erilaisista lähtökohdista, eli jos tuotannossa tultaisiin ensisijaisesti hyödyntämään toista sovellusta, eikä Altéa FM -ohjelmistoa, testaus suoritettiin ohjelmistointegraatioiden toteutusta vasten, mutta tietojen siirtymisen osalta testitapaukset oli validoitava Altéa FM -sovelluksessa. Jos tuotannossa loppukäyttäjä tuli taas käyttämään esimerkiksi tiedon syöttämiseen suoraan Altéa FM -ohjelmistoa, tehtiin tällöin testaus ohjelmistossa, eli varsinaisessa ohjelmistomoduulissa. Useamman moduulin osalta

tilanne oli kuitenkin se, että tuotannossa voitiin käyttää sekä erillistä ohjelmistoa, joka integroitiin viestiliikenteen avulla osaksi Altéa FM -ohjelmiston toiminnallisuuksia että Altéa FM -ohjelmiston moduuleita. Tällöin ohjelmiston testaus tuli tehdä sekä Altéa FM -ohjelmistomoduulissa, mutta myös ohjelmistojen integraatioiden toimivuus piti verifioida ja validoida, jotta sen käyttö olisi tarpeen tullen mahdollista. Tämän pohjalta ohjelmiston testaus voitiin jakaa koko projektissa perustestaukseen. Tämä kokonaisuus toteutettiin ohjelmistomäärityksiin pohjautuen ja käyttötapauksiin pohjautuen, jotka muodostettiin liiketoimintaosaamisen ja todellisten skenaarioiden avulla. Lisäksi testattiin integraatioiden toteutukset ja näistä järjestelmistä siirretyn datan käytön mahdollisuus Altéa FM -ohjelmistossa. Yhden moduulin testauksen osalta oli kyse muustakin kuin vain ohjelmiston toiminnallisuuksien verifioinnista. Testitapausten kirjaamisen avuksi toteutettu mallidokumentti löytyy liitteestä 5.

4.6.2 Rahtimoduuli

Testitapauksien yleiskuvaus ja esimerkkejä

Eri lentokonetyypeille lähetetään rahdin järjestelmästä rahtitieto ja verifioidaan, että oikeat tiedot tulevat Altéa FM -ohjelmistoon perille ja väärät tiedot eivät välity järjestelmään. Tämä operaatio testattiin manuaalisesti ja sen lisäksi rahtitietojen lähetys tehtiin myös toisesta järjestelmästä. Testattavana olivat esimerkiksi seuraavanlaiset skenaariot: konteilla lastattavaan koneeseen yritettiin lähettää viestiä, joka sisälsi ylipainoisia kontteja, viestissä oli täytettynä myös ylimääräinen vapaa tietokenttä, ennakkotiedot sisältävä viesti ja lopullinen rahdin viestitieto lentoa varten eri konetyypeille, sekä niin sanottu tyhjä rahtitieto, eli tieto, ettei rahtia jollekin lennolle ole lainkaan.

Testitulokset ja havainnot tiivistettynä edellisten esimerkkien pohjalta

Testauksessa havaittiin muun muassa, että vapaan tekstikentän tieto ei tullut lainkaan aluksi testattaessa perille erityisesti, jos kyse oli viestiliikenteen avulla toimitettu tieto. Tämä tieto ei jostain syystä siirtynyt sille tarkoitettuun kenttään Altéa FM -sovelluksessa. Havaitsimme myös testattaessa, että jos vaarallisen aineen tietoja oli merkittävästi esimerkiksi ruumakohtaisesti tai konttikohtaisesti, niin näiden tarkistaminen oli työlästä. Käyttöliittymästä kyseessä olevan ruuman tiedot piti avata erilliseen ikkunaan, jotta kaikki tiedot saatiin näytölle. Ennakkotietojen osalta niiden päivittäminen oli helposti mahdollista uuden lähetetyn viestin avulla, mutta lopullisen rahtitiedon päivittäminen piti suorittaa Altéa FM -sovelluksesta, jos se jostain syystä oli virheellinen. Määrityksen mukaan lopullista rahtitietoa ei voi enää lähettää uudelleen, jolloin siihen liittyvät korjaukset piti hoitaa suoraan Altéa FM -järjestelmän kautta tai lennon valmistelijan kanssa yhteistyössä. Jos lennolle ei tullut lainkaan rahtia, eli

lähetettiin niin sanottu ”Final Nil” -viesti, ilmeni, ettei tätä viestin käsittelyä ja ominaisuutta ollut toteutettu lainkaan Altéa FM -sovelluksessa huomioon. Amadeuksen oli toteutettava järjestelmään tämä ominaisuus, jonka avulla ymmärretään tyhjän rahtitiedon informaatio myös suoraan standardiviestistä, sillä joka lennolle ei ole rahtia.

Ylipainoisten konttien osalta viesti tulee perille, mutta kontti ei allokoidu koneeseen, koska kontin paino on liian suuri. Järjestelmä toimi tässä tapauksessa oikein ja varoittaa kontista. Lennon valmistelijan on tässä tapauksessa puututtava tilanteeseen sen ratkaisemiseksi. Testitapaus, jota myös testattiin, oli väärien konttityyppien syöttäminen lennolle, jolle ne eivät käytännössä mahdu, koska kontit ovat vääränkokoisia. Tällöin ohjelma hylkäsi myös nämä viestit, koska kyseessä oleva konttityyppi ei sovellu koneeseen, johon sitä on yritetty tarjota. Rahtimoduulissa myös havaittiin testauksen aikana sellainen puute, että rahdin moduulista puuttui nappi, jolla pystyi poistamaan esimerkiksi virheellisen rahtikontin tai rahtitiedon, eli käytännössä käyttöliittymästä rivin listauksesta. Tämä toiminnallisuus toteutettiin Amadeuksen toimesta käyttöliittymään ilman muutoshallintaa, sillä kyseessä oli määrityksessä mainittu ominaisuus, mutta puute varsinaisessa ohjelmistossa.

4.6.3 Kuormausmoduuli

Testitapauksien yleiskuvaus ja esimerkkejä

Testauksessa suoritettiin testauksia sekä hiiren avulla että ilman hiirtä. Selkeitä testitapauksia olivat liikkuminen käyttöliittymässä, konttien siirto ruumasta toiseen, matkatavaran siirto kontista irtoruumaan, kontin numeroiden ja lopullisten painojen syöttö ja vaarallisten aineiden tietojen päivitys. Lisäksi oli testattava, että kuittaustoiminnallisuus kuormauksesta ja lennon valmistelusta toimii oikein kumpaankin suuntaan. Jos kuormauksen kuittausta ei tehdä oikeassa järjestyksessä tai lennon valmistelija ei hyväksy kuormaa, niin mahdollisten muutoksien jälkeen kunkin lennon kuormaus pitää kuitata taas uudelleen sekä kuormauksesta että lennon valmistelusta vastaavan tahon toimesta.

Testitulokset ja havainnot tiivistettynä edellisten esimerkkien pohjalta

Kuormausmoduulin toiminnallisuudet toimivat kohtalaisen hyvin, mutta siellä oli vakavia puutteita käytettäessä moduulia näppäimistön avulla. Hiiren avulla moduulia oli helppo hyödyntää, mutta näppäimistöä pelkästään hyödynnettäessä toiminnallisuuksille ei yksinkertaisuudessaan ollut koodattu vastaavia näppäinkomentoja. Vaikeuksia oli lähinnä käyttöliittymän eri osioiden välisessä siirtymisessä, jolloin muun muassa siirtyminen eri taulukkojen sarakkeiden, rivien ja eri kenttien välillä ei oikeastaan toiminut. Joihinkin tietoihin ei päässyt käsiksi lainkaan ilman hiirtä, jolloin toiminnallisuuksien suorittaminen ei ollut mahdollista ilman hiirtä. Nämä puutteet

korjattiin vaatimuksien perusteella, mutta lopullinen toiminnallisuus ei palvellut hyvin loppukäyttäjiä, joiden työskentely sijoittuu yleensä ulkotiloihin. Kuormausmoduulin kuittauksen testaus oli myös tärkeä testattava, sillä kun kuormaajat ovat kuormanneet koneen, tulee heidän ilmoittaa lennon valmistelijalle lopullinen kuormaus, erityisesti, jos suunniteltuun kuormaukseen on tullut muutoksia. Hiukan hiontaa prosessimielessä vaatinut toiminnallisuus päätettiin varmuuden vuoksi vielä vahvistaa erillisellä viestillä ohjelmistossa, jotta kuormauksen valmistumisesta ei synny epäselvyyttä. Ongelmalliseksi asian teki se, että jos lennon valmistelija ei hyväksynyt kuormaa kuormaushenkilöstön ilmoittamalla tavalla, tuli hänen ottaa yhteyttä kuormaajaan ja ohjeistaa mahdollisen kuormauksen muutokset. Ensisijaisesti kuormausmoduulin testauksen lähtökohtana oli suunnitella ja toteuttaa testitapaukset mahdollisimman lähelle totuutta vastaaviksi eri konetyypeille, jotta Altéa FM -ohjelmiston käyttö kuormausmoduulin osalta olisi mahdollista.

4.6.4 Lentopetrolin määrä

Testitapauksien yleiskuvaus ja esimerkkejä

Polttoaineen osalta tärkeimmät testitapaukset olivat ennakkoon kullekin lentoreitille suunnitellun lentopetrolin määrän siirtyminen toisesta järjestelmästä Altéa FM -järjestelmään ja tämän tiedon päivittyminen tarvittaessa lennon uusia tasapainolaskelmia varten. Järjestelmä, josta tieto siirtyy Altéa FM -ohjelmistoon, on tarkoitettu lennon reitityksen suunnitteluun ja näin sen avulla on mahdollista optimoida eri tekijät huomioiden lennolle tarvittava lentopetrolin määrä mahdollisimman lähelle optimaalista määrää. Lennon suunnittelun aikana muuttuvien arvojen tuli myös siirtyä Altéa FM -järjestelmästä takaisin reitin suunnittelujärjestelmään. Tämän tiedon perusteella lennon reittisuunnittelu pystyi vielä mahdollisuuksien mukaan päivittämään lentopetrolin määrää esimerkiksi annettujen tietojen pohjalta tai reitin mahdollisiin muutoksiin liittyen. Vielä ennen lennon lähtöä mahdolliset viime hetken muutokset ja tiedon päivitys lopullisen polttoaineen määrän osalta täytyi myös saada siirtymään Altéa FM -ohjelmistoon. Tämän tiedon kulkua varten koneen kapteenille annettiin monta mahdollisuutta tiedon päivittämiseen.

Testitulokset ja havainnot tiivistettynä edellisten esimerkkien pohjalta

Polttoaine oli mahdollista syöttää suoraan Altéa FM -ohjelmistoon omasta moduulista tästä vastuussa olevan osaston henkilöstön toimesta. Sen lisäksi tieto voidaan lähettää järjestelmästä, jolla lentopetrolin määrää kullekin reitille suunnitellaan. Integraation toiminnallisuuden varmistaminen oli tässä haastavin kokonaisuus, eli testaus keskittyi toteutetun viestiliikenteen varmistamiseen kahden järjestelmän välillä. Lisäksi Altéa FM -ohjelmistossa testattiin polttoaineen erilaisten määrien vaikutus lentokoneen

tasapainoon ja sen sijoittelu koneessa eri konetyypeillä. Lentopetrolin määrään liittyy tietynlaisia lainalaisuuksia, joiden täytyy tietyillä konetyypeillä toteutua, jotta konetta voidaan kuormata tai lentää. Sen takia eri lentokoneen petroolitankkien käyttäminen ja näiden eri variaatioiden testaaminen ohjelmistossa oli tärkeää. Testauksessa myös tarkistettiin, että niin sanotut turvarajat ja ennalta asetetut asetukset toimivat eri konetyyppien kanssa oikein. Haastavaksi osoittautui lentopetrolin viime hetken tietojen päivittäminen. Viime hetken tietojen päivityksessä ohjeistettiin lennon kapteeni toimittamaan tieto suoraan koneesta yhdellä valitsemallaan tavalla lennon valmistelijalle ja Altéa FM -ohjelmistoon.

4.6.5 Matkustajien ja matkatavaroiden lähtöselvitys

Testitapauksien yleiskuvaus ja esimerkkejä

Matkustajien lähtöselvityksessä testitapauksia oli selkeästi eniten ja tämä kokonaisuus oli Altéa FM -ohjelmiston käyttöönotossa haastavin kokonaisuus. Ensinnäkin tämä kokonaisuuden hallinta tuli myöhemmin suoritettavaksi Altéa CM -ohjelmiston avulla, jonka käyttöönotto suoritettiin Altéa FM -ohjelmiston jälkeen ja välimenovaiheen ajaksi piti toteuttaa ohjelmistointegraatio vanhasta lähtöselvitysympäristöstä Altéa FM -ohjelmistoon. Käytännössä piti testata sekä matkustajien että laukkutietojen siirtyminen Altéa FM -sovellukseen ja tätä ennen toteuttaa tarvittavat viestimuutokset Finnairin käyttämään lähtöselvitysjärjestelmään. Testitapausten suorituksen ohella tuli myös varmistaa, että viestiliikenne ja integraatio Altéa FM -ohjelmiston ja AY FIDO -ohjelmiston kanssa toimii moitteettomasti. Tämän kehitystyön suunnitteluun, toteuttamiseen ja testaamiseen kului projektissa oma sille suunniteltu aikansa ja tämä kehitystyö oli kahden Finnairin palkkaaman ohjelmistokehittäjän vastuulla.

Matkustajien ja matkatavaroiden lähtöselvitykseen liittyen testitapauksia oli 80 - 110 laskukaavasta riippuen. Matkustajatietojen osalta eri testitapaukset piti myös verifioida Altéa FM -ohjelmistossa ja tämän lisäksi testata näiden toiminnallisuudet Altéa FM -ohjelmistossa. Matkustajatietoihin liittyen erilaisia testitapauksia oli erilaisille matkustajille, matkustajaluokille, konetyypeille, ryhmille, lapsille ja sylilapsille. Matkatavaroiden osalta testattiin myös erilaiset testitapaukset, eli ruumaan menevä matkatavara, käsimatkatavara, erikoismatkavara, myöhässä oleva matkatavara ja näihin liittyvät poikkeustapaukset. Matkustajien lähtöselvitykseen kuuluu myös lennon eri tiloihin liittyvät tilamuutokset lennon osalta ja ohjelmistossa ja näiden testaaminen yhdessä lennon valmistelijan näytön kanssa. Matkustajien lähtöselvityksessä testattiin kaikki Finnairin matkustajaprosessissa tunnistetut erikoistapaukset, kuten lemmikki lennolla, erityisruokavaliot ja istumatoiveet tai matkustajaluokan korotus. Lisäksi Finnairille tyypillinen vapaa istumajärjestys koneessa kotimaan lennoilla vaati muutoshallintaa, jotta tämä toiminnallisuus oli ylipäänsä mahdollista toteuttaa. Matkustajien lähtöselvitykseen liittyen testattiin myös niin sanottu

linkin tai yhteyden katkaisu AY FIDO:n ja Altéa FM:n välillä. Tällöin matkustajatietojen päivitys oli mahdollista suoraan Altéa FM -ohjelmistosta käsin ja tiedot eivät enää tällöin siirtyneet AY FIDO:sta Altéa FM -ohjelmistoon. Tämä skenaario saattoi toteutua esimerkiksi, kun lento oli jo suljettu lähtöselvityksestä ja matkustajien koneeseen otto portilta aloitettiin, mutta jos esimerkiksi tapahtuu niin, että kaikki koneeseen lähtöselvitetyt matkustajat saavu portille ajoissa ja syystä tai toisesta myöhästyvät koneesta.

Testitulokset ja havainnot tiivistettynä edellisten esimerkkien pohjalta

Matkustajatietojen siirtymiseen liittyvät testitapaukset olivat vaativia toteuttaa, sillä testaukseen tarvittiin kahta toisistaan riippumatonta järjestelmää, eli aikaisemmin mainitut AY FIDO - ja Altéa FM -ohjelmisto. Tämän takia testausta varten tuli suunnitella testitapauksien lähdedata ja asetukset kumpaankin järjestelmään. Kaikki matkustajien lähtöselvitykseen ja porttitoimintoihin liittyvät toiminnallisuudet toimivat vanhassa AY FIDO -lähtöselvitysjärjestelmässä ja tiedon siirto järjestelmien välillä kaikkien eri skenaarioiden osalta piti verifioida ja validoida myöhemmin tuotannon beta-versiossa. Matkustajien lähtöselvitykseen liittyen avattiin koko projektissa eniten virheraportteja ja myös muutoksenhallinnan kautta lisätöitä. Vapaavalintaisen istumajärjestyksen mahdollistamiseksi, erikoismatkatavarat ja näiden sijoittelu ruumassa, sekä henkilökunnan matkustaja-asetusten ja sääntöjen noudattamiseksi avattiin muutoshallinnan kautta lisätyöpyynnöt. Vaativimpia testitapauksia olivat jatkoreitityksiin liittyvät testitapaukset ja lentokoneen ruumiin lemmikkieläimien sijoittelu perustuen ennalta asetettuihin sääntöihin. Merkittävästi aikaa meni integraation toteutuksen aikana tapahtuvaan testaukseen, eli samojen testitapausten toistamiseen yhä uudelleen kehitysvaiheessa.

Altéa FM -ohjelmiston osalta perusnäyttö ja -moduuli matkustajatietojen osalta toimivat luotettavalla tasolla, mutta lennon eri tilojen kanssa oli useampia ongelmia. AY FIDO -ohjelmistosta lennon tilaan vaikuttava tieto ei muuttanut lennon tilaa Altéa FM -ohjelmistossa tai se muuttui väärin. Linkityksen katkaisu näiden kahden ohjelmiston välillä vaikutti myös siihen, että tämän tapahtuessa tuli lennon valmistelijoille ja muille matkustajalähtöselvityksessä mukana oleville laatia selkeät ohjeet toimintatapojen osalta. Tunnistetun käyttötapauksen poikkeuksena tässä oli huomioitava, etteivät Altéa FM -ohjelmistossa tehdyt muutokset enää päivittäneet AY FIDO -ohjelmistoa linkin katkaisun jälkeen. Myös matkustajien lisätietoihin liittyen tietojen siirtymisessä oli puutteita, eli erikoisruokavalion tilaukset toimitettiin varmuuden vuoksi myös ateriapalveluun, jotta tieto erikoisruokavalioista oli vähintään kahdessa järjestelmässä. Matkatavaratietojen siirtymisen kanssa oli myös aluksi ongelmia, mutta viestien formaattimuutokset korjasivat nämä ongelmat. Asetuksien säätäminen myös lentokonekohtaisesti takasi sen, että jotkin testitapaukset saatiin suoritettua onnistuneesti, sillä osoittautui, että itse asiassa asettamissamme asetuksissa oli puutteita. Sylilasten ja ryhmävarausten testitapausten testauksen yhteydessä osoittautui, että

joidenkin lentokonetyyppien osalta asetukset eivät olleet täysin kunnossa. Tiettyyn konetyyppiin voidaan ottaa sylilapsen lepopaikalle vain tietty määrä sylilapsia, jonka jälkeen heille on aina varattava oma istumapaikka. Ryhmien lähtöselvityksen osalta ilmeni sellainen seikka, etteivät ryhmäläiset aluksi sijoittuneet koneeseen ryhmän mukana, jos ryhmässä oli tietyn määrän ylittävä lukumäärä matkustajia. Ryhmäkohtaisia matkustaja-asetuksia muutettiin ja asetuksia korjattiin myös tiettyjen konetyyppien asetuksiin. Matkustajaluokkien muutokset ja matkustajaluokan korotus tuottivat myös omat haasteensa. Tasapainolaskelman osalta merkittävin seikka matkustajien lähtöselvityksen ja koneeseen oton osalta olivat matkustajan sukupuoli ja ikä ja sitä myöten matkustajalle määräytyvä paino järjestelmään. Matkustajien sukupuoli ja painot osaltaan vaikuttavat tasapainolaskelmien toteuttamiseen. Sen lisäksi matkustajien istumakartta on tärkeässä osassa lennon lähtöön liittyvien arvojen määrittelyssä ja näiden sääntöjen osalta tehtiin myös tietyiltä osin hienosäätöä joidenkin konetyyppien ja erilaisen matkustajamäärän osalta.

4.6.6 Lennon valmistelu ja lentojen tasapainolaskelmien toteuttamiseen tarkoitettu moduuli ja lennon valmistelijan toiminnallisuudet

Testitapauksien yleiskuvaus ja esimerkkejä

Lentojen valmisteluun liittyviä testejä oli valmisteltava myös muille Altéa-ohjelmistoille, sillä eri rajoitukset tai asetukset lentokoneen painojen osalta syötetään Altéa Admin -sovelluksen kautta. Lentojen siirtymiseen Altéa Inventorystä Altéa FM -ohjelmistoon vaikuttavat erilaiset asetukset, joita on esitelty liitteessä 1: Altéa-ohjelmistoperheen ohjelmistot ja niihin liittyvät avaintoiminnallisuudet. Lennon valmistelijalla saattoi olla myös esimiesrooli, jolloin hän myös pystyi toimimaan muiden lennon valmistelijoiden esimiehenä ja allokoimaan lennot henkilökunnalle, tarkastamaan sertifikaatit muiden työntekijöiden osalta. Perustestaukseen lennon valmistelijan roolissa kuuluivat automaattisen kuormausohjeen toteuttaminen, viestiliikenteen lähetys, saapuminen ja testaaminen eri toimijoiden kesken, hyväksymiskäytännöt tai kuittaukset ja näiden testaaminen, asetettujen hälytyksien tai viestien toimitus Altéa FM -ohjelmiston toimesta ja kaikkien moduulien avulla toimitettujen tietojen päivittyminen lennon valmistelijan päänäytölle Altéa FM -ohjelmistossa.

Lennon valmistelijan testitapauksien rajaus tehtiin siten, että kaikki toiminnallisuudet, jotka olivat jo testattu jonkin moduulin testaajan toimesta ja eri järjestelmien ohjelmistointegraatiot, rajattiin pois tästä kokonaisuudesta. Käytännössä testitapaukset kirjoitettiin kaikille muille toiminnallisuuksille Altéa FM -ohjelmiston osalta, joita ei vielä aikaisemmin ollut testattu. Lennon valmistelijan osalta testaus jouduttiin tekemään myös erilaisia käyttäjärooleja silmällä pitäen, eli näitä olivat lennon valmistelija, esimies ja lennon valmistelun keskuksen esimies. Testitapaukset saattoivat

vaihdella laidasta laitaan perustestauksesta spesifisempään lennon tasapainolaskelman arvojen tarkasteluun. Edellä mainittuja testejä kuvastaa kaksi keskeistä seikkaa, joita ovat kokonaisprosessin valvonta ja viestiminen eri tahoille tarvittaessa ja lennon tietojen oikeellisuuden tarkastelu Altéa FM -ohjelmistosta ja tarvittaessa poikkeamiin puuttuminen.

Testitulokset ja havainnot tiivistettynä edellisten esimerkkien pohjalta

Testitapausten tulosten perusteella hienosäädettiin asetuksia ja rajoja erilaisille tasapainolaskelmien ylä- ja alarajoille. Testauksessa painottui ensisijaisesti koneen erilaisten kuormien vaikutuksien tarkastelu tasapainolaskelmien avulla ohjelman avulla saataviin arvoihin. Jokaiselle konetyypeille tehtiin myös automaattisia kuormaukseen liittyviä sääntöjä, jolloin lennon valmistelijalle oli yksinkertaisesti Altéa FM -ohjelmistossa nappia painamalla toteuttaa koneen kuormausohjeistus näiden ennakkosäätöjen avulla. Testauksessa myös korostui eri näyttöjen käyttö lennon valmistelijan moduulista ja näiden rinnakkainen käyttö. Lisäksi eri lennon tilojen muutokset ja näiden erilaiset hälytyksien aiheuttamat hälytystyypit tuli testattua tässä kokonaisuudessa ja näiden kanssa tehtiin myös edelleen hienosäätöä.

Lennon valmistelijan moduulin testitapauksina oli myös tarkistaa, että eri ohjelmistoista, kuten Altéa Inventory - ja Altéa Admin -ohjelmistoista sinne asetettujen tietojen nojalla lennot syntyivät Altéa FM -järjestelmään oikeine tietoineen. Näitä tietoja olivat esimerkiksi lennon konetyyppi, koneen rekisterinumero ja muut koneeseen liittyvät konekohtaiset asetukset ja tämän lisäksi kullekin reitille ominaiset reittitiedot. Katso tarvittaessa Altéa-ohjelmistoperheen kaavio liitteestä 1, jossa on kuvattu kunkin ohjelmiston avaintoiminnallisuudet. Tulevaisuudessa lentokohtaisesti oli myös mahdollista saada historiatietoon perustuvat arvot tasapainolaskelmien pohjatiedoiksi.

Lennon valmistelijan testaukseen liitettiin myös Altéa FM -ohjelmiston viestimoduulin testaus ja sen asetuksien validointi myös tuotannossa. Viestimoduuli oli periaatteessa oma kokonaisuutensa, mutta sen testaus sidottiin lennon valmistelijan testitapauksiin, koska viestimoduulin pääasiallinen käyttö tapahtui lennon valmistelijan toimesta. Näin testaajien oli myös tarkistettava, että kaikkien viestipohjien formaatti täsmäsi vaatimusten mukaisesti. Viestiformaatin osalta jouduttiin myös avaamaan muutoshallinnan avulla lisätyöpyyntö, jotta kuormauksen raportointi (Load Sheet) ja siihen liittyvän dokumentaation saatiin vastaamaan Finnairilla käytössä olevaa formaattia. Muutoshallinnan avulla toteutetun muutoksen avulla dokumentti sisälsi oikeat arvot, joita koneen lentoon lähtöä varten tarvittiin Finnairilla.

4.6.7 Testitapauksien kirjoittaminen

Jokainen järjestelmäasiantuntija vastasi omasta osa-alueestaan, moduulista ja sen ensisijaisesta testaamisesta. Näitä olivat aikaisemmin mainitut rahtimoduuli,

matkatavaramoduuli, matkustajatietojen moduuli ja lentopetrolin informaatiomoduuli sekä kaiken datan keräävä kuormalaskelman toteuttajan näyttö. Jokaisen järjestelmäkehittäjän vastuulla oli tarkistaa yhden osa-alueen testitapaukset ja antaa näistä mahdolliset kommentit tai kehitysehdotukset. Testaajien ammattitaidossa korostui liiketoimintaosaaminen tai syväosaaminen omalta ammattialaltaan ja testauspuolesta ei ollut testitapauksia kirjoittaessa merkittävää osaamista (Stenberg 2007a, s. 22). Testitapaukset perustuivat pitkälti perusolettamukseen, miten ohjelmiston pitäisi toimia ja käyttötapausten tunnistamiseen (Stenberg 2007a, s. 20). Vaikka projektissa oli käytössä ohjelmistomoduulien määritysdokumentaatio oli tärkeää tunnistaa sellaiset käyttötapaukset, joita ohjelmistoa suunniteltaessa ei Amadeuksella osattu ottaa huomioon ja jotka olivat Finnairille ehdottomia vaatimuksia, joita ilman tuotantokäyttöön ei voitu siirtyä. Yksi esimerkki tällaisesta toiminnallisuudesta oli koneen lähtöselvitys ilman matkustajalle merkittyä istuinnumeroa, mikä oli Finnairille ehdoton edellytys kotimaan reiteillä. Esimerkki testitapausten dokumentointiin liittyvästä taulukosta löytyy liitteestä 5.

4.7 Muut testaustekniikat, joita projektissa hyödynnettiin

Tässä kohdassa esittelen muita projektissa käytettyjä testaustapoja, jotka poikkeavat ohjelmiston perustestaamisesta ja joiden avulla pyrimme virheettömään lopputulokseen. Lisäksi mukana on testaamista, mikä suoritettiin, kun uusia ominaisuuksia otettiin käyttöön tai tarvittiin esimerkiksi jonkin kokonaisuuden tai toiminnallisuuksien uudelleen testaamista. Tämän muotoinen regressiotestaus suoritettiin uuden tuotantoversion julkaisun yhteydessä, jolloin testataan regressiomaisesti ohjelmiston merkittävimmät ja tärkeimmät käyttötapaukset ja yleensä myös ohjelmistokehittäjien korjaamat virhekorjaukset tai uudet ominaisuudet ja niiden toimivuus.

4.7.1 Satunnaistestaus

Satunnaistestaukseen valittiin arpomalla tiettyjä toiminnallisuuksia kaikista testitapauksista ja näitä suoritettiin tietyiltä osin jo kertaalleen testattujen ominaisuuksien testaamisen osalta. Esimerkiksi satunnaistestejä suoritettiin uusien ohjelmistoversioiden päivityksien jälkeen beta- ja testiympäristössä. Näin saatiin satunnaisesti todennettua, että muutokset eivät vaikuttaneet eri moduuleihin tai ohjelmiston toiminnallisuuksiin. Tässä lähestymistavassa lähdettiin siitä, että koska kaikkea ei voi testata, niin satunnaistestauksen avulla edes osa toiminnallisuuksista saadaan testattua (Taina 2007, s. 5). Satunnaistestausta käytettiin hyödyksi regressiotestauksessa, josta voi lukea alakohdassa 4.7.5.

4.7.2 Järjestelmäintegraatioiden testaaminen ja ohjelmistojen välisen viestiliikenteen toimivuuden testaaminen

Erillisten järjestelmäintegraatioiden testaus suoritettiin niiden toteutuksen jälkeen Finnairin ja sen yhteystyökumppaneiden käytössä olevilla sovelluksilla, joiden piti toimia yhdessä Altéa FM -ohjelmiston kanssa. Vain näin saatiin aikaiseksi toimiva kokonaisuus, jolloin Altéa FM -ohjelmistoa voitaisiin täysipainoisesti hyödyntää. Joidenkin integraatioiden osalta tehtiin testausta myös erillisessä integrointitestiympäristössä, minne viestien lähettäminen oli mahdollista häiritsemättä muuta ohjelmistotestausta, eli testaus tehtiin omassa erillisessä kehitysympäristössä. Lähettävän ohjelmiston testiympäristö, josta viesti toimitettiin integrointitestille ja kun testi oli tämän osalta mennyt läpi. Integraation toimivuus voitiin vielä tarkastaa varsinaisessa testiympäristössä ja sitten edelleen tuotannon beta-versiossa. Järjestelmien keskinäisen viestiliikenteen toimivuus ja sen varmistaminen on avainasemassa kokonaissysteemin järjestelmäintegraatiotestauksessa (Jääskeläinen, Katara & Vuorio 2012, s. 118). Viestin tutkiminen tai ohjelmistointegroinnin verifiointi vaati vähintään toimittajan ja järjestelmäasiantuntijoiden yhteystyötä reaaliaikaisesti, jotta mahdollinen viestin tai tietojen siirtyminen pystyttiin varmistamaan eri ympäristöjen osalta. Testausajat oli sovittava yleensä ennalta useamman tahon kesken, jotta saimme testit onnistuneet suoritettua ja tarkistettua testien lopputuloksen, eli viestin siirtymisen Altéa FM -ohjelmistoon. Testauksen ajan pidettiin reaaliaikainen yhteys puhelimitse tai viestintäsovelluksella eri tahoihin, jotta mahdollinen viestijärjestelmään liittyvä selvitystyö oli mahdollista reaaliajassa.

Haasteena järjestelmäintegraatioiden ja viestiliikenteen testaamisessa eri järjestelmien välillä oli se, ettei esimerkiksi testiympäristöstä voinut lainkaan toimittaa viestejä suoraan tuotannon beta-versioon tai tuotantoon. Testaus piti suorittaa ensin testillä ja myös tuotannossa esimerkiksi kuvitteellisilla lennoilla, jotka missään nimessä eivät tulisi siirtymään varsinaiseen reittiliikenteen listoihin. Vaikkakin lentojen reitit olivat sellaisia, joita Finnair ei todellisuudessa lennä, laitettiin lentosarjat sellaisille lentonumeroille, jotka on valittu testilentonumeroiksi ja ne eivät tulleet näkyviin normaaleihin päiväkohtaisiin lentolistoihin. Toinen haaste testauksessa oli osoitteisto, joihin viestit piti toimittaa ja avaukset näiden osoitteiden osalta. Esimerkiksi Amadeuksen joihinkin viestiliikenteen osoitteisiin ei voitu lähettää viestejä kuin hyväksytyistä ja tunnistetuista osoitteista. Kaikki avaukset ja asetukset piti olla kunnossa ennen viestin lähettämistä. Tiettyjen viestiformaattien osalta oli myös havaittavissa selkeitä puutteita, eli kaikkien standardiviestien formaatteja ei ollut käyty Amadeuksen toimesta läpi ja nämä viestit eivät tämän takia saapuneet perille. Usein ennen testausta oli parasta ensin tarkistaa Amadeuksen vastuuhenkilöiden kanssa vaaditut viestit ja niiden formaatti merkki merkiltä, jotta viesti oli mahdollista vastaanottaa ja sen tarkistus oli varmasti toteutettu oikein Altéa FM -ohjelmistossa. Lisätietoa ohjelmistojen välisestä systeemitestauksesta on mahdollista lukea kohdasta 4.6 ja sen alakohdista.

4.7.3 End-to-end testaus, eli ohjelmiston kokonaisvaltainen testaaminen

Hyväksymistestauksen osalta Finnairilla oli määritelty tietyt kriteerit ja tämä näkyi myös koko projektin ajan siten, että sovellus lopullisesti voitiin hyväksyä käyttöönotettavaksi. Amadeukselle oli tärkeää todentaa koko ohjelmiston toimivuus Finnairin tarpeisiin ja tarvittavien muiden ohjelmistojen kanssa toimivaksi erillisellä end-to-end -testauksella. Näin voitiin varmistaa koko ohjelmiston käyttö kaikilla normaaleimmilla käyttötapauksilla lennon aktivoinnista aina sen arkistointiin saakka ja paikantaa mahdolliset ongelmat kokonaisuuden osalta. Tämä testaus valmisteltiin erittäin hyvin, koska se oli niin kokonaisvaltainen ja monia henkilöitä koskettava, että sen osalta ei haluttua joutua tilanteeseen, jossa jokin asia estäisi prosessin läpiviennin ohjelmiston ja henkilöstön kanssa. Lisäksi testattavana olivat erilaisten skenaarioiden lennot, jotta käyttötapauksiltaan testaus olisi mahdollisimman kattava. Myös tietyt poikkeustapaukset testattiin, kuten takaisin kääntyvä lento. Merkittävää tässä testaustavassa on se, että jos jostakin järjestelmästä ei tieto saavu Altéa FM -ohjelmistoon tai toisaalla suoritettu toiminto päivitä jotain tarvittavaa tietoa. Tämä näkyy heti testin aikana ja tällöin suoritetaan korjaustoiminto suoraan esimerkiksi Altéa FM -sovellukseen manuaalisesti. Tällaisen mahdolliset poikkeavuudet kirjattiin ylös ja koko testisessio purettiin heti testauksen päätyttyä purkutilaisuudessa testaukseen osallistuneiden osapuolten kanssa. Tällöin myös sovittiin toimenpiteistä mahdollisten esteiden poistamiseksi järjestelmän ja prosessin toimivuuden takaamiseksi.

Kyseessä on tarkemmin koko ohjelmiston käyttö kokonaisvaltaisesti sillä laajuudella tuotannon beta-versiossa erilaisten toiminnallisuuksien osalta, kuin miten se tuli Finnairille käyttöön myös lopullisessa tuotannossa. Testaukseen osallistuivat viimeisessä vaiheessa myös oikeasti lennon osalta kaikki ne tahot, jotka normaalisti myös syöttävät tai tuovat lennolle tarvittavaa dataa. End-to-end -testauksella on tarkoitus käydä koko ohjelmiston toiminnallisuudet läpi, jotka siihen liittyvät ja näin todeta koko prosessin toimivan (Roodenrijs 2009). Gensin (2011, s. 68) mukaan end-to-end -testaaminen mahdollistaa ohjelmiston käyttöönoton testauksessa ohjelmiston toimintojen laajan testauksen koko ohjelmiston osalta lyhyessä ajassa tehokkaasti. Tähän pyrittiin myös Finnairin end-to-end -testauksessa ja myöhemmin ohjelmiston rinnakkaiskäytöllä valittujen lentojen osalta.

4.7.4 Parallel-testaus, eli rinnakkaistestaus

Projektin aikana testauksen lähtödatana ja myös laajempien testiskenaarioiden pohjana käytettiin reaalidataa oikeista esimerkeistä, mitä syntyi päivittäin Finnairin lentojen pohjalta. Olemassa olevasta tuotantojärjestelmästä kopioitiin kaikki tarvittava data, joka liittyi johonkin ennalta valittuun lentoon ja testaus pyrittiin toteuttamaan samalla tavalla uuden Altéa FM -ohjelmiston ja Altéa FM -ohjelmistoon integroitujen muiden ohjelmistojen kanssa testiympäristössä. Parallel-testauksessa, eli kahden järjestelmän rinnakkain käyttämisessä testaus suoritetaan tarkalleen sillä tavalla, kuin se tuotannossa

kullakin hetkellä tapahtuu. Testiohjelmistoa tai tuotannon beta-versiota käytettiin tuotannossa olevien järjestelmien kanssa rinnakkain ja suoritetaan testaajien toimesta samoja toimintoja kuin tuotannossa. Testaus suoritettiin samanaikaisesti reaaliajassa mukailleen tuotannossa tapahtuvia toimenpiteitä ja niiden etenemistä. Testaukseen valittiin myös poikkeustapauksia lennon käännön osalta ja testaaminen suoritettiin testiympäristössä seuraavana päivänä samoilla arvoilla ja toimenpiteillä kuin tuotannossa oli toteutunut. Näin voitiin testauksessa havainnoida, miten uusi, tai testauksessa oleva järjestelmä käyttäytyy esimerkiksi koko lennon elinkaaren osalta. (Hetzel 1998, s. 140-141.)

Altéa-projektissa rinnakkaistestausta käytettiin toteamaan järjestelmän toimivuus pelkästään Altéa FM -ohjelmistoa käytettäessä, eli ilman integroitujen ohjelmistojen syötteitä. Testaus toteutettiin myös yhdessä integroitujen ohjelmistojen kanssa, jolloin testijärjestelmistä syötettiin dataa eri järjestelmien avulla Altéa FM -ohjelmistoon. Varsinainen rinnakkaistestaus tehtiin uuden järjestelmän kanssa tuotannossa lentojen avulla, jotka eivät näkyneet oikeassa tuotantoympäristössä, vielä ennen virallista ja varsinaista siirtymistä oikeaan tuotantokäyttöön. Tämä testaus suoritettiin järjestelmän loppukäyttäjien toimesta lentokentillä, CLC-keskuksessa, projektihenkilöstön toimesta ja muissa valituissa kohteissa, jolloin ohjelmiston käyttöä oli mahdollista testata myös mahdollisimman laajasti lopullisessa ohjelmiston käyttöympäristössä. Rinnakkaistestaus osoitti heikkoudet tiettyjen toiminnallisuuksien osalta ja esimerkiksi, jos sähkeiden toimittamisessa tai tulostuksessa oli ongelmia. Tämä testitapa oli myös erittäin hyvä harjoitus maapalveluiden edustajille uuden järjestelmän käyttöön, sillä testausta tuettiin myös projektihenkilöstön toimesta. Testaus myös osoitti ne kohdat, joiden osalta oli vielä hiottavaa prosessin kannalta tai lisäkoulutuksen tarvetta.

4.7.5 Regressiotestaus ja ohjelmistoon korjattujen toiminnallisuuksien uudelleen testaaminen

Regressiotestaus suoritettiin projektin aikana alihankinnan avulla uusien tuotannon beta-versioiden julkaisun yhteydessä. Tämän testauksen osalta toteutettiin myös testauksen automatisointia, jotta testaus oli mahdollista suorittaa samojen testitapausten osalta useamman kerran tehokkaasti ja nopeasti. Aivan kaikkia ohjelmiston toiminnallisuuksia ei testattu uuden tuotannon ohjelmistoversion julkaisun yhteydessä, vaan tietty kokonaisuus rajattiin testitapauksiksi, jotka suoritettiin kunkin ohjelmistoversion julkaisun yhteydessä perustoiminnallisuuksien osalta. Kokonaisuus saattoi myös vaihdella siten, että eri tuotantoversioiden julkaisun yhteydessä testattiin kolmessa eri syklissä tietyt kokonaisuudet kattavammin kuin muut ominaisuudet. Regressiotestauksella pyritään toteamaan, että korjaukset eivät toimi tai jo korjatuissa toiminnallisuuksissa on edelleen virheitä (Jääskeläinen, Katara & Vuori 2012, s. 267). Regressiotestauksessa käytetään uudelleen vanhoja testitapauksia ja testataan nämä testitapaukset uudelleen jo aikaisemmin testattujen ohjelmiston toiminnallisuuksien ja

kokonaisuuksien osalta. Regressiotestauksessa usein testataan ominaisuuksia, jotka on jo testattu, ja pyritään löytämään näistä edelleen virheitä. Tuomikosken (2009, s. 14) mukaan regressiotestauksen lähtökohtana on tarkistaa, ettei kertaalleen jo toimiva ominaisuus hajoaa tai lakkaa toimimasta esimerkiksi uusien toiminnallisuuksien implementoinnin takia. Esimerkiksi, jos vanhaan toiminnallisuuteen liitetään jokin uusi toiminnallisuus tai ominaisuus, pyritään osoittamaan, että vanha toiminnallisuus edelleen toimii vaikka siihen on tuotu uusia ominaisuuksia (Tervonen 2009, s. 17).

Projektin yksi tarkoitus oli koota niin kattava testitapausten joukko ja dokumentaatio, että Altéa-projektin päätyttyä tarvittavat testaukset voitiin suorittaa joko alihankkijan toimesta tai sitten automatisoinnin avulla. Uusien ohjelmistoversioiden testaaminen ei jäänyt täysin Finnairin sisäiseksi tehtäväksi, sillä siihen tullaan tulevaisuudessa käyttämään ohjelmistoalan osaajia, jotka testit suorittavat. Näiden valvontaa ja testauksen ohjausta hoidetaan edelleen tulevaisuudessa Finnairilta tai Amadeuksen toimesta. Uusien toiminnallisuuksien testausvastuu ja käyttöönotto määritellään tulevaisuudessa tapauskohtaisesti.

5 TESTAUSPROSESSI, PROSESSIN DOKUMENTOINTI JA TESTAUKSEN KEHITTÄMINEN PROJEKTISSA

Hyvä perusta prosessin kehittämiseksi on dokumentaation tuottaminen ja testauksen tason ja sen tavoitteiden määrittäminen. Lähtökohtana tälle oli Finnairilla ensisijaisesti projektin onnistunut läpivienti, testausprosessin dokumentointi jatkuvaa testausta varten esimerkiksi regressiotestausta varten ja lisäksi prosessin ja testauksen jatkuva kehittäminen. Finnairilla on lukuisia ohjelmistoja käytössä, mutta näiden hallittu ja dokumentoitu testaaminen oli tällä tasolla aikaisemmin jäänyt lähinnä toimittajien ja alihankkijoiden vastuulle. Stenbergin (2007b, s. 2) mukaan testausprosessia voidaan pitää kuvauksena testauksen aikana tuotettavasta dokumentaatiosta. Tämä käsittää esimerkiksi testaussuunnitelman, testauksen suoritukseen liittyvän dokumentaation, kuten testiraportit ja muun testauksen dokumentaation. Pelkästään näillä dokumenteilla ei kuitenkaan vielä voida päästä testauksen prosessin hallintaan ja laadukkaaseen lopputulokseen testauspalveluiden osalta vaikkakin Bach, Kaner & Pettichord (2002, s. 233) kirjoittavat testaussuunnitelman istuttamisen kulloiseenkin tarpeeseen ja tavoitteiden mukaiseen laajuuteen olisi riittävää. Tähän tarvitaan kuitenkin aikaa ja prosessin kehittämistä, testauksen kirjaamista osaksi laatukäsikirjaa ja tapaa toimia, sekä ohjeistusta eri testauksen tasojen saavuttamiseksi. Näin voidaan taata testauksen kehittäminen, laadun parantaminen ja prosessin kehittäminen. Bach et al. (2002, s. 234) kuvaavat asiaa prosessin kehittämiseen tarvittavan viisi eri lähtökohtaa, eli mission, tavoitteet tai vaatimukset, testauskäytännöt ja niiden vakiinnuttamisen, testauksen toteuttavat tahot ja jatkuvan kehittämisen. Testauksen prosessin kehittämistä voidaan laajentaa myös strategisessa mielessä tärkeäksi osaksi toimintaa, että se itsessään selittää testauksen tarpeellisuuden (Bach et al. 2002, s. 236). Finnairilla Altéa-projektissa prosessin kehittäminen nähtiin niin tärkeäksi ja myös testaus osana kokonaisprojektia, että projektissa haluttiin panostaa laadukkaaseen lopputulokseen.

Testaustapa projektissa muistutti pitkälti testauksen V-mallia (Haikala & Märijärvi 2002, s. 287), missä toteutettiin periaatteessa peruskoodin -toiminnallisuuksien ja ohjelmiston perustestaaminen, moduulitestaaminen moduuleittain, moduulien ja ohjelmiston integraation testaus yhdessä. Lisäksi testattiin integraatioiden toteutukset osana ohjelmistomoduulien toimintoja. Lopuksi suoritettiin erilaisia laitetestauksia ja rinnakkaistestaus tuotantojärjestelmän kanssa ja ennen tuotantoon käyttöönottoa myös koko ohjelmiston toiminnallisuuksien mukainen järjestelmätestaus end-to-end -testauksena.

5.1 Testausprosessin osa-alueet Altéa-projektissa peilaten TMMi-mallin tasoon 2: hallinta

Projektin aikana testausprosessi kehittyi ajan myötä ja käytännöt vakiintuivat testitapausten dokumentoinnin osalta. Tämä taas tarkoitti sitä, että testauksen ja dokumentoinnin toteutukseen tuli rutiininomaisuutta, joka näkyi testitapausten suorittamisen nopeampana läpimenoaikana ja dokumentoinnin laadun parantumisena. Arvioni pohjalta pääsimme TMMi:n tasolle kolme (toimintatavat) ja myös testauksen prosessin kehittäminen oli mahdollista testauksen rinnalla projektin aikana. TMMi:n tasolle neljä (mittaaminen) on kuitenkin niin spesifisiä ohjelmistoliiketoimintaan ja testauksen liittyviä toimintoja, että niiden saavuttaminen olisi vaatinut testauksen kokonaisuuden ottamista osaksi koko kaupallisen ryhmän toimintaa. Toisaalta saavutimme myös tasolta neljä osan osa-alueen vaatimuksista ja tärkeintä oli jatkuva prosessin kehittäminen ja testauksen arvostuksen nostaminen ja näin myös mahdollisuus kokonaislaadun parantamiseen, jotka ovat TMMi:n tason viisi (kehittäminen) kokonaisuuksia.

Testauksen ja käyttöönottoon projektin aluksi toteutettiin testausstrategia, jonka sisällysluettelo löytyy liitteestä 7: Altéa-projektissa käytetyn testausstrategian sisällysluettelo. Strategiassa kuvattiin toimintatavat ja testauksen tavoite ja laajuus sekä testien jakaantuminen eri ajanjaksoille. Testausstrategiassa listattiin myös erilaiset testaustavat. Lisää strategiasta ja sen laajuudesta voit lukea kohdasta 4.2 testauksen strategia. TMMi-mallin toisen tason (hallinta) vaatimuksiin strategian dokumentointi kuuluu olennaisena osana ja tämä dokumentti tuli projektissa toteutettua ja ylläpidettyä. Strategian ohella laadittiin testaussuunnitelma, jossa tarkemmin otettiin kantaa testauksen suorittamiseen ja operatiiviseen toimintaan projektissa testauksen osalta. Testauksen suunnitteluvaiheessa ja testaussuunnitelman avulla on mahdollista tarkasti kuvata, että miten testaus suoritetaan (Hetzel 1998, s. 124). Testien suunnittelu ja testitapauksien dokumentointi ja varsinainen testaus taas keskittyy enemmän itse testauksen suorittamiseen. Drabickin (2004, s. 75) mukaan testausprosessin mallinnusta ja suunnittelua voi tehdä testaussuunnitelman toteutuksen aikana ja tällöin pohti, että missä kohdin prosessissa on parannettavaa tai minkä kokonaisuuden kehittämiseen on hyvä keskittyä. Testaussuunnitelman sisältö on IEEE-829 standardin mukaisesti seuraavanlainen (van Veenendaal 2012, s. 41):

- testaussuunnitelman tunnus
- yleiskuvas testauksesta
- mahdolliset poikkeamat tai tarkennukset suhteessa testausstrategiaan
- testattavat ja ei-testattavat ominaisuudet ja kokonaisuudet
- toiminnallisuudet, jotka testataan tai ei testata
- testaustekniikat
- hyväksymiskriteerit
- keskeyttämisen ja uudelleen testauksen kriteerit

- testauksen aikataulutus ja testauksen aikana vaaditut dokumentit
- testauksen elinkaari ja päätehtävät projektissa
- testausympäristö ja vaatimukset
- resursointi, vastuut ja mahdolliset koulutukset
- sidosryhmien sitoutuminen ja osallistuminen
- testitapausten laajuus karkealla arviolla
- testausaikataulu ja
- testausprojektin riskit sekä tunnistetut ongelmat.

Testaussuunnitelman sisällysluettelo on liitteenä 8: Altéa-projektissa käytetty testaussuunnitelman sisällysluettelo, josta voi nähdä millä laajuudella Altéa-projektissa testaussuunnitelma toteutettiin. Altéa-projektissa testauksen suunnittelun osalta oli erityisesti seuraavien asioiden huomioiminen testaussuunnitelmassa tärkeää:

- Testaussuunnitelmaa päivitettiin tarpeen tullen ja kummallekin ohjelmiston testaukselle tehtiin oma suunnitelma. Käyttöönottoa varten ja hyväksymistestaukselle toteutettiin omat erilliset dokumentit. Lentokenttien testaussuunnitelma ja tuotannon validointia varten tehtiin omat dokumentit. Tällä tavalla toimittaessa saatiin kokonaisuudet ositettua selkeästi.

- Testaussuunnitelman toteutus integraatioiden osalta oli tärkeässä osassa testaussuunnitelmaa ja näiden toteutukseen ja testauksen suunnitteluun panostettiin merkittävästi, sillä ilman Finnairin lähtöselvitysjärjestelmän integrointia koko Altéa FM -ohjelmistoa ei olisi voinut ottaa käyttöön.

- Testaussuunnitelma tarkastettiin muiden projektiin kuuluvien henkilöiden toimesta ja myös hyväksyttiin projektin ohjausryhmässä.

- Testaussuunnitelma käytiin läpi projektin viikkopalaverissa, jotta sen sisältö, tarkoitus ja jalkautus olivat mahdollista projektiin osallistuville. Testaussuunnitelman avulla dokumentoitiin myös kunkin osa-alueen vastuut ja vastuualueiden varahenkilöt. Viikkopalavereissa pidettiin myös kirjaa juoksevista tehtävistä, jotka piti toteuttaa projektin aikana ja resursoitiin työtehtäviä tarpeen tullen, jos esimerkiksi testauksen etenemisen seurannassa kävi ilmi poikkeamia aikataulusta.

- Testauksen aikataulutus oli tärkeää, jotta voitiin sopia ja suunnitella muita projektiin liittyvä toimenpiteitä, kuten koulutusvalmistelut ja koulutuksien ajankohdat loppukäyttäjille. Aikataulutukseen osalta kriittiseksi muodostuivat muutoshallinnan kautta avatut

toiminnallisuuksien muutokset, koska ne piti toteuttaa ennen varsinaista käyttöönottoa.

Vaikka Altéa-projektissa ei projektin aikana ollut käytössä tiettyä prosessiin kehittämiseen tähtäävää mallia, niin silti saavutimme tietyt suuntaviivat testauksen käytäntöjen osalta. Projektin aikana syntyneen testauksen prosessin kehittymistä on ollut mahdollista arvioida jälkikäteen mm. TMMi-mallin avulla. Ajan ja projektin edetessä testauksen käytännöt ja prosessi paranivat ja jatkuva kehittäminen ja uudistuvat toimintatavat olivat osa projektia. Projektissa ei suoraan sovellettu testauksen prosessin kehittämiseen ja dokumenttien parantamiseen tarkoitettua mallia tai mallinnuspohjaa. Testitapausten ja testauksen suorittamisen parannuksiin suorittamamme toimenpiteet voidaan nähdä toteutuneen pitkälti Drabickin (2004, s. 95-97) esittämän testausprosessin mallin kehittämisen kaavion pohjalta. Projektin aikana käytössä olivat testauksen suorittamiseksi tietyn tyyppiset pohjadokumentit, joihin testitapaukset kuvattiin perustuen tiedossa oleviin käyttötapauksiin ja kirjattiin toivottu testin lopputulos. Dokumentit myös katselmoi joku muu kuin dokumentin toteuttaja. Dokumentilla oli vähintään toinen silmäpari tarkistamassa dokumentin ja testitapausten oikeellisuutta. Testidata piti toteuttaa suunnitelmallisesti tiettyjen ajanjaksojen välillä, jotta testit voitiin suorittaa. Testitapaukset merkittiin testatuksi tietyin käytännöin ja muutenkin sovitusta merkintätavoista pidettiin kiinni sen osalta, jos testi ei mennyt läpi tai siihen liittyi mahdollinen muutospyyntö. Edellä mainitut toimenpiteet on Drabick (2004, s. 96) kuvannut testausprosessin mallissa, mutta hänellä on lisäksi mukana itse testattava ohjelmisto ja korjattu ohjelmistoversio. Nämä olivat projektissa olennaisena osana testausta ja joka viikko testaus tehtiin uusimmalla mahdollisella ohjelmiston testiversiolla. Drabickin (2004, s. 96-97) testauksen mallinnuksessa ovat avainasemassa prosessiin sisään tuotavat asiat tai tiedot ja sieltä ulos tulevat asiat, kuten esimerkiksi dokumentit.

Eri kokonaisuuksien osalta kustakin testitapaukset sisältävästä MS Excelistä testauksen tilanne raportoitiin viikoittain koontiexceliin, jonka perusteella saatiin viikoittaiset testauksen etenemisen tunnusluvut. Katso koontiexcelin malli liitteestä 6. Näin voitiin sekä arvioida testauksen tuloksia että mahdollisia valmistumisajankohtaa kuin myös testaukseen sisältyviä kustannuksia. Projektissa oli TMMi:n tason kaksi, eli hallinnan osalta, myös testaukseen suunnittelun ja toteutuksen ohella myös testauksen monitorointi ja hallinta mukana projektin aikana. Testit suoritettiin kunkin testaajan omalla koneella, mutta sen lisäksi projektilla oli käytössä testilaboratoriossa laitteistot, joita lentokentillä oli tarjolla.

5.2 TMMi-mallin tason 3 ja 4 arviointi testausprosessin näkökannalta Altéa-projektissa

TMMi:n tason kolme, eli toimintatapojen osalta pystyttiin täyttämään kaikki osa-alueet melko hyvin tältä tasolta Altéa-projektissa. Projektissa oli testausorganisaatio ja Altéa-projektiin sovelletut roolit projektin etenemisen aikana muuttuivat tarpeen mukaisesti. Liitteessä 3 ja 4 on esitelty Altéa-projektin henkilöstö ja tarkemmin Altéa CM -projektiin osallistuneet henkilöt ja sidosryhmät. TMMi:n tasolla kolme vaadittu koulutusohjelma projektin henkilöstölle oli tässä tapauksessa riittävä, sillä kaikki ohjelmistojen toiminnallisuudet koulutettiin kaikille testaajille ja sen lisäksi testauksessa käytettävien järjestelmien ja laitteistojen koulutusta tarjottiin. Koulutus ei sinänsä projektissa ollut jatkuvaa tai tiettyä vuosisykliä toteuttavaa, mutta riitti projektin laajuuden osalta hyvin. Lisäkoulutusta oli myös saatavilla esimerkiksi ohjelmistointegraatioihin liittyen ja näiden järjestelmien käyttöä varten. Myös muutokset projektin dokumentoinnissa raportoitiin projektin henkilöstölle ja tarvittaessa käytiin läpi projektin viikkopalavereissa. Testauksen ja testausprosessin kehittäminen takasi projektin aikana sen, että testauskäytännöt pyrittiin vakiinnuttamaan osaksi projektin ja Finnairin kaupallisen ryhmän toimintaa vastaavissa projekteissa. Tähän oli mahdollisuus Altéa-projektin jälkeen, koska toimintatapojen, sekä projektin dokumentointiin käytettiin merkittävästi aikaa. Dokumenttien mallipohjat olivat koko kaupallisen ryhmän saatavilla, mutta manuaali dokumentaatiosta ei projektin aikana testausprosessin hallintaa varten toteutettu. Näitä oli mahdollista hyödyntää tarvittaessa myös tulevaisuudessa tarpeen mukaan.

Ei-toiminnallinen testaus oli mukana testauksen laajuudessa, kuin se nähtiin tarpeelliseksi käyttöönoton osalta. Testaus suoritettiin asennusten onnistumisen osalta eri ympäristöihin ja ohjelmistojen toimivuuden tarkistaminen tehtiin erilaisilla näytön resoluutioilla. Katselmointi keskittyi lähinnä dokumenttien katselmointiin ja niiden kierrättämiseen tarvittavilla tahoilla, sekä uuden ohjelmistoversioiden toiminnallisuuksien katselmointiin. Katselmointipalavereita käytettiin myös muutoshallinnan ominaisuuksien läpikäymiseen, jotta toiminnallisuuden määritys ja vaatimukset olivat Amadeukselle ja Finnairille selkeitä ja näiden perusteella pystyttiin toteuttamaan selkeät käyttötapaukset. Tämä oli ensisijaisen tärkeää sekä toimittajalle että Finnairille, sillä osa muutoshallinnan toiminnallisuuksista nousi käyttöä estäviksi vaatimuksiksi vasta parisen kuukautta ennen käyttöönottoa. Varsinaisia katselmointeja testauksen tilanteesta ei kuitenkaan testauksen näkökulmasta pidetty vaan testauksen seurantaan ja siihen liittyvien puutteellisten ominaisuuksien läpikäynti tapahtui projektipalavereissa 2-3 viikon välein. Palavereissa oli omina osioinaan testauksen etenemisen ja tilanteen seuranta, ongelmaraporttien läpikäyminen, muutoshallinta sekä mahdollisten epäselvyyksien läpikäynti. Amadeuksella oli katselmointien osalta kirjattu tietyt käytännöt laatukäsikirjaan ja katselmointitilaisuudet järjestettiin Altéa-projektin

johdon kesken uuden tuotantoversion julkaisun yhteydessä (tuotannon beta-versio tai varsinainen tuotantoversio).

TMMi:n tasolla neljä mittaamisen on avainasemassa. Mittaaminen tietyllä tasolla toteutui Altéa-projektin aikana, mutta koska kyseessä oli vain yksi projekti kahden erillisen ohjelmiston käyttöönotossa, ei mittaamisen käytäntöjä tullut viikoittaista seurantaa paremmin kehitettyä projektin aikana. Mittaristo lähinnä vertaili testauksen edistymistä edellisiin viikkoihin nähden, mutta sen avulla ei tehty vertailua tai optimointeja eri projektien kesken. Tämän tason saavuttaminen ja mittariston kehittäminen ohjelmistotuotteiden arviointiin on tärkeää, jotta laadunhallintaa voidaan myös kehittää. Altéa-projektissa tähän ei kuitenkaan lähdetty. Käyttöönotettavien ohjelmistojen piti täyttää Finnairin asettamat käyttötapaukset ja toiminnalliset vaatimukset vaikka osittain prosesseja lennon käännössä ja lähtöselvityksessä pyrittiin projektin aikana muuttamaan. Laadullisten kriteereiden tuli vastata Finnairin sille asettamia vaatimuksia. Laaduntarkkailu kuului projektissa kuitenkin enemmän toimittajan vastuulle ja laatukriteereitä ei varsinaisesti määritelty Altéa-projektissa tarkemmin testauksen osalta. Tärkeää oli saada testaus määritetyn laajuuden mukaisesti hyväksytysti suoritettua aikataulun puitteissa, jotka projektille oli asetettu. Edistyneiden vertaiskatselmusten tarve olisi varmasti tulevaisuudessa myös Altéa-tuotteiden regressiotestauksen arvioinnin osalta hyvä ottaa käyttöön, mutta projektin aikana tässä vertailua ei erikseen lähdetty määrittämään ja toteuttamaan.

5.3 Testauksen hallinta dokumentaation avulla

Altéa-projektin ohjauksen ja testaukseen liittyvään yleiseen suunnitteluun ja testauksen hallintaa varten toteutettiin Altéa-projektissa esimerkiksi seuraavat dokumentit:

- testausstrategia (liite 7 sisältää sisällysluettelon tästä)
- testaussuunnitelmat eri osaprojekteille ja kokonaisuuksille (liite 8 sisältää sisällysluettelon tästä, mitä sovellettiin Altéa-projektissa)
- projektisuunnitelma ja aikataulu
- ohjelmistomäärityket eri moduuleista, joita käytettiin apuna testitapausten suunnittelussa
- testitapaukset ja näiden raportointi suoritetun testauksen pohjalta moduuleittain ja ohjelmistointegraatioiden osalta (katso liite 5)
- muutoshallinnan dokumentointi ja näihin liittyvät käyttö- ja testitapaukset
- testauksen raportointi viikoittain (liite 6)
- asennuskohteiden rekisteri ja asennuksien status tuotannossa
- laitteistojen validointisuunnitelma ja raportointilomake
- end-to-end -testaussuunnitelma
- käyttöönottosuunnitelmat eri osaprojekteille ja kokonaisuuksille

- käyttöönoton tukidokumentaatio sekä
- käyttöohjeet ja manuaalit.

Näiden edellä mainittujen dokumenttien avulla testauksen etenemistä voitiin seurata ja edelleen kehittää testauksen prosessin kehittämistä ja myös dokumentaation laatua parantaa. Teimme Altéa-projektissa testauksen hallinnasta tarpeisiimme sopivan laajuisen. Avainasemassa oli testauksen etenemisen seuranta testauksen suorituksen perusteella ja perusdokumentaation täydentäminen tarpeen tullen. Lisäksi projektin eri vaiheissa toteutettiin erilaisia dokumentteja, jos jokin asia ja tieto oli tarpeen jakaa useamman tahon kesken ja ennen kaikkea, jotta asiat saatiin sovittua ja dokumentoitua myös tulevia tarpeita varten.

Dokumentoinnin merkitystä ei voida kiistää, sillä sen avulla voidaan todeta suunniteltujen testitapauksien testaustulokset ja uudelleen toteuttaa ne tarkalleen, niin kuin ne ensimmäisellä kerralla toteutettiin. Tarvittaessa oli mahdollista todeta moduulikohtaisesti, miten testit on suoritettu ja millaisiin tuloksiin niissä on päästy. Uusien ominaisuuksien testaaminen oli periaatteessa mahdollista keneltä vain järjestelmäasiantuntijalta ja esimerkiksi alihankintana ostettuna palveluna. Lisäksi dokumentaatio testausprosessin, testisuunnitelmien ja raporttien osalta jää Finnairille mahdollisuuksien mukaan edelleen käytettäväksi ja jatkokehittämistä varten. Dokumentointi takasi jatkotestauksen mahdollistamisen helposti, koska dokumentit olivat samassa formaatissa järjestelmäasiantuntijasta tai testaajasta riippumatta. Näin ohjelmisto on periaatteessa mahdollista ottaa uudelleen testattavaksi ja regressiotestauksen toteuttaminen uusien ohjelmistoversioiden osalta on myös mahdollista.

Lähtökohtaisesti testausdokumenttien merkitys oli testausprosessin kehittämisen osalta erittäin merkittävä. Jos yhtään dokumenttia ei olisi toteutettu, näitä dokumentteja arvioitu ja edelleen paranneltu, niin silloin testauksen prosessia ei olisi voinut kehittää. Dokumentit ja niiden toteuttaminen takasi myös niiden kehittämisen ja testauksen etenemisen seurannan. Jokaisen oli projektin testauksen dokumentoinnin osalta noudatettava siihen asetettuja ohjeistuksia ja tapoja toteuttaa dokumentaatiota eri vaiheissa. Tähän aluksi liittyi haasteita, mutta asian osalta päästiin melko nopeasti yhteisymmärrykseen projektin koko henkilöstön kanssa, kun asian todettiin pidemmällä aikavälillä helpottavan testauksen toteuttamista ja hyödyttävän koko projektin läpiviemistä.

Sen sijaan käytössämme ei ollut konfiguraation hallintaan liittyvä varsinaista dokumentaatiota, koska tämän puolen hoiti ohjelmistotuotteen osalta Amadeus. Dokumenttien viimeiset versiot säilytettiin yhteisessä projektitilassa, johon kaikilla projektin osallisilla oli pääsy. Sen lisäksi uudet versiot dokumenteista nimettiin ennalta sovituin säännöin. Stenbergin (2007c, s. 6) mukaan testauksen hallinnalla tarkoitetaan ensisijaisesti testitapausten suunnittelua ja toteuttamista, tulosten analysointia ja mahdollisten lopetuskriteereiden arviointia. Lähtökohtaisesti hallinnan avulla on pitää nämä eri toimenpiteet kontrollissa ja valvoa, että projekti ja testaaminen etenevät

suunnitellun mukaisesti. Altéa-projektissa oli tärkeää hallita sekä projektin etenemistä kokonaisuudessaan että testauksen hallintaa, joka muodosti tärkeän kokonaisuuden Altéa-projekin käyttöönotosta. Tärkeimpiä työvälineitä olivat testaukseen varsinaisiksi työdokumenteiksi toteutetut testitapausten dokumentaatio ja raportointidokumentit (katso liite 5) ja testauksen raportointiin toteutettu testauksen etenemistä mittaava ja raportoiva dokumentti (liite 6: viikkoraportoinnin malli Altéa-projektissa). Raportoinnin pohjalta testauksen hallintaan kuuluu raporttien tarkastelu, myös testauksen priorisointi ja kontrollointi (Stenberg, 2007c, s. 18-19). Erityisesti, kuten dokumentaation määrästäkin voidaan päätellä, testauksen hallintaan kuuluu suunnitelmallisuus, eli miten jokin asia toteutetaan ja tämän dokumentointi sekä toteutuneen edistymisen raportointi ja edistymisen valvonta. Vastuussa tästä oli Altéa-projektissa kaikki testaajat, mutta avainasemassa oli testauksen koordinointivastuullinen projektin testausvastaava.

5.4 Testauksen suorittaminen ja testauksen prosessin kehittyminen

Testauksen suorittamisesta, testitapausten suunnittelemisesta, dokumentoinnista, sen tarkastamisesta eli katselmoinnista ja tulosten kirjaamisesta on jo kirjoitettu luvun 4 kohdissa 4.3.–4.7. Katso lisäksi tarvittaessa liite 5, eli malli testaustaulukosta, johon sekä testitapaukset että tulokset kirjattiin. Taulukkopohjaa käytettiin hyödyksi kunkin kokonaisuuden ja ohjelmistomoduulin testaamisessa. Näiden dokumenttien ensimmäinen aste oli lähinnä taulukko, jossa kukin testitapaus karkeasti kuvattiin, mutta projektin loppuvaiheissa testitapausten ja tulosten dokumentilla oli seuraavat ominaisuudet:

- etusivu
- virallinen nimi ja versionumero
- katselmoijan kommentit
- sisällysluettelo
- testitapauksiin liittyvät dokumentit
- erilaiset testitapaukset eroteltuna korkeampien kategorioiden mukaisesti ohjelmistomoduulin päätoimintojen mukaisesti
- kehittynyt virheraportointi
- mahdolliset virheraportit ja niiden kuvaus ja
- moduulista mahdollisesti avattu muutoshallinta.

Virheraportointi suoritettiin Amadeuksen tarjoamalla virheidenraportointijärjestelmällä, jonne kirjattiin virheeseen liittyvät vaaditut tiedot, kuten mistä sovelluksesta virhe löytyi, mistä versiosta ja mistä moduulista ja kelle se annettiin ratkaistavaksi ja tietenkin virheen kuvaus sekä mahdolliset lisätiedot, kuten lokitiedosto ja ruutukaappaukset.

Eri kokonaisuudet ja ohjelmistomoduulit jaettiin omiksi kokonaisuuksikseen ja näille kaikille kokonaisuuksille nimettiin vastuuhenkilö suorittamaan testaus. Näin tehtiin myös ohjelmistointegraatioiden, muutoshallinnan ja erilaisten testaustapojen osalta, joita olivat esimerkiksi end-to-end -testaus ja regressiotestaus. Testauksen tulokset ja eteneminen kultakin viikolta koottiin viikkoraporttiin (katso tarvittaessa liite 6), missä näkyi suoraan esimerkiksi seuraavat seikat:

- suoritettujen testitapausten tilanne kokonaisuuksittain, eli Altéa FM -ohjelmiston moduulit, ohjelmistointegraatiot ja muutoshallinta
- tavoitepäivä, jolloin testaus on kaikkien testitapausten osalta suoritettu
- suunniteltujen testitapausten määrä
- testauksen tilanne
- prosenttisuhde testatuista testitapauksista suhteessa suunniteltuihin
- virheraporttien määrä viikkotasolla ja kokonaisuudessaan
- testauksen jatkumista estävät virheraportit ja tarkempi kuvaus virheistä ja
- vastuuhenkilö.

Tämän raportin avulla voitiin viikkotasolla seurata testauksen etenemistä ja virheiden määrän kehittymistä, sekä pidemmällä aikavälillä ja trendiä testauksen edistymisestä. Raportin avulla voitiin myös kohdentaa testauksen prioriteettia siirtämällä sellaisia kokonaisuuksia testaukseen, mitkä eivät edenneet odotetusti. Etenemisen trendin ja kaavioiden avulla, sekä luvatun päivityssyklin avulla voitiin ennustaa, milloin ollaan siinä tilanteessa, että voidaan siirtyä testaamaan uusia toiminnallisuuksia tai seuraavaa testausmuotoa, kuten parallel-testausta tai end-to-end -testausta. Osa testeistä piti suorittaa usealla eri konetyypillä eli useita kertoja ja edelleen näiden suorittaminen myös tuotannossa lopullista validointia varten oli tärkeää. Joidenkin testitapausten suorittaminen ja varmistaminen tuotannossa oli myös tärkeää projektin loppuvaiheilla tiukkojen aikataulujen takia. Suurin osa toiminnallisuuksista testattiin myös rinnakkaistestauksen aikana ja myöhemmin end-to-end -testauksen aikana laajojen käyttötapauksien suorittamisen aikana.

Olisimme voineet käyttää käyttötapausten, testitapausten, dokumentoinnin, testauksen hallinnan ja testauksen raportoinnin osalta myös siihen suunniteltua ohjelmistoa, jolloin kaikki materiaali olisi ollut yhdessä ohjelmistossa tai paikassa. Projektissa oli kuitenkin oleellista saada dokumentointi testauksen osalta kuntoon mahdollisimman yksinkertaisesti ja toteuttaa se välineillä, joiden käyttö oli yleisluonteisesti mahdollista ajasta ja paikasta riippumatta. Päädyimme Microsoftin perussovellusten, kuten taulukkolaskennan, eli Excelin, ja tekstinkäsittelyohjelmiston, eli Wordin käyttöön.

5.5 Testausprosessin kehittämisen haasteet

Testauksen aikana ei ollut käytössä lähdekoodia, joten tältä osalta testaus piti suorittaa mustan laatikon periaatteella. Toisaalta oman arvioni mukaan tällä ei juuri olisi ollut lisäarvoa, sillä harvalla Altéa-projektiin osallistuvilla oli minkäänlaista ohjelmointikokemusta. Näin koodin lukeminen olisi ollut melkoisen turha toimenpide. Lisäksi testaus suoritettiin melko pitkälti liiketoiminnan ja Finnairin toimintatapojen näkökulmasta vaikka myös testausta tehtiin Altéa FM -ohjelmiston perustoiminnallisuuksien kanssa. Toisaalta määritysten lukeminen ja niihin tutustuminen oli myös haastavaa ohjelmistotuotantoa tuntemattomille henkilöille. Dokumentaatioon tutustuminen antoi kuitenkin osviittaa siitä, miten ohjelmiston ja moduulien pitäisi toimia ja millaisia arvoja ohjelmistoon voidaan syöttää. Määritysten tarkka lukeminen myös takasi sen, että Amadeukselta eli toimittajalta voitiin vaatia niitä toiminnallisuuksia ohjelmistolle, jotka dokumentteihin oli kirjattu, mutta syystä tai toisesta nämä olivat jääneet toteuttamatta. Finnairin lähtökohtana testaukselle oli melko pitkälti liiketoiminnoille kriittisten toiminnallisuuksien toimivuuden tarkistaminen toimiviksi tai vastaavasti ja myös muutoshallinnan kautta avattujen toiminnallisuuksien testaaminen.

Projektissa vaikutti aluksi myös kokemattomuus testauksessa ja testauksen raportoinnissa. Vahvin osa-alue oli testitapausten ja myös käyttötapausten dokumentointi, joka perustui kunkin järjestelmäasiantuntijan aikaisempaan asiantuntemukseen kustakin lentokoneen lähtöön liittyvästä kokonaisuudesta. Raportointia parannettiin heti ensimmäisten testitapausten listauksien valmistuttua kustakin moduulista listaamalla suunnitellut testitapaukset, paljonko näistä on testattu, millaisia virheraportteja on avattu ja mikä oli kunkin testaajan oma arvio testauksen valmistumisesta kutakin kokonaisuutta kohden. Raportointia kehitettiin edelleen Altéa FM -projektin ja erityisesti Altéa CM -projektin aikana, kuten tämänkin työn ja liitteiden perusteella voi havaita. Lähtökohtaisesti oli tärkeää saada raportoinnista mittaustuloksia käytettäväksi testauksen etenemisen ja päättämisen ennustamiseen ja mahdollisten resurssien kohdentamiseen. Vasta testauksen raportoinnin kehittäminen toi myös testaajien vastuun kantamisen kunnolla esille projektissa, sillä tavoitteisiin pääseminen tai niissä epäonnistuminen näkyi melko nopeasti viikoittaisessa raportoinnissa. Dokumentaation tuottamista myös muilla osa-alueilla parannettiin Altéa-projektin loppua kohden, jolloin myös tavoitteet, mahdolliset riskit ja aikataulu sekä vastuut olivat selkeämmin jokaisen projektiin osallistuvan tiedossa. Dokumenttien saatavuus ja lävitse käyminen niiden valmistuttua toi projektiin selkeyttä ja myös ohjaus ja projektin johto tätä myötä tehostui.

Uusien järjestelmien kanssa työskentely oli myös aluksi haastavaa, sillä järjestelmiä ei osattu kunnolla käyttää tai niiden käytön opettelu vei runsaasti aikaa. Eräänlainen haaste liittyi viikoittain ladattavaan uuteen Altéa FM -ohjelmiston testiversioon, joka alkuaikoina unohtui asentaa, ennen kuin tästä tuli viikoittainen rutiini. Suunnittelusta huolimatta myös testeissä tarvittavan raakadatan ja testidatan syöttämisessä oli

haasteita, kunnes suunnitelmallisella ja järjestelmällisellä toimintatavalla tämä kokonaisuus saatiin kuntoon. Varmistimme datan kopioinnin ajankohtana, että vastuu testidatan syöttämisestä oli viikkotasolla allokoitu jollekin henkilölle ja tarvittaessa hänen varahenkilölleen.

Projektin kokonaishallintaan ylipäänsä sisältyi haasteita sen takia, että projektin osapuolia oli ympäri Eurooppaa ja myös Intiassa. Loppuasennukset piti hoitaa myös Japaniin ja Yhdysvaltoihin ja Kanadaan. Merkittävimmät haasteet tässä globaalissa projektissa tuli kielimuurin ja erilaisten toimintakulttuurien kohdatessa. Myös projektinhallinnassa oli kulttuurieroja eri painopistealueiden kanssa, johtuen hiukan erilaisista projektimalleista ja testauskäytännöissä. Lähtökohtaisesti sitouduimme asiakkaana noudattamaan Amadeuksen tarjoamaa projektimallia ja toimittaja antoi merkittävää tukea esimerkiksi dokumentaation, parhaiden käytäntöjen ja asennuksien käyttöönoton kanssa. Testauksen osalta arvokasta osaamista ja tukea saimme taas ITC Infotechilta, mikä myös vastasi osaltaan testaamisesta. Yhteistyö intialaisten kanssa oli paikoittain haasteellista kielen, aikaeron ja toimintatapojen takia, mutta myös erilainen kulttuuri ja työtavat poikkesivat suomalaisista. Haasteellisinta oli alihankinnan valvonta, jonka takia päädyimme siihen, että yksi ITC Infotechin testauskoordinaattori istui koko Altéa-projektin testausvaiheen ajan kanssamme samassa toimipisteessä Vantaalla. Näin hän pystyi tarkemmin valvomaan intialaisia kollegoitaan ja me pystyimme vaatimaan häneltä tavoitteemme mukaisten kokonaisuuksien edistymistä tietyllä aikataululla. Etäpalaverikäytäntöihin sisältyi myös omat haasteensa, vaikka etäpalavereiden järjestämisessä oli käytettävissä uusin teknologia. Lentoyhtiöllä oli myös se etu, että myös normaaleja kokouksia oli suhteellisen helppoa ja edullista järjestää Euroopan sisällä. Tämä oli yksi avain projektin edistämisessä, kun joidenkin asioiden kanssa tuntui olevan suurempia haasteita kuin toisten. Amadeuksen henkilöstön oli myös helppo tulla Helsinkiin tukemaan projektin tiettyjä vaiheita tarpeen mukaan tai ennalta sovittuina ajankohtina.

TMMi:n mallin arvioinnin suoritin itsenäisesti teoriaan ja kirjallisuuteen, eli lähdemateriaaliin, perustuen. TAMAR-arviointimenetelmän soveltamisen osalta ei myöskään dokumentoitu varsinaiseen arvioimiseen liittyvää suunnitelmaa, sillä lähinnä tämä työn osuus oli osana diplomityötäni. Projektin ja testauksen aikana ei lähtökohtaisesti toimittu saavuttaaksemme jonkin testausprosessin kehitykseen liittyvän kypsyystason. Varsinainen prosessin kehittämisen tavoite projektissa oli se, että kehittämisen avulla projektin läpiviemistä oli mahdollista tehostaa. Testauksen läpiviemisessä ja testausprosessin kehittämisessä oli apua niin alihankkijasta, toimittajasta kuin Finnairilla kollegoistakin. Arvioimiseen tarvitsi myös perspektiiviä ja varsinainen arvioinnin suorittaminen itse projektin aikana ei olisi antanut yhtä kypsää kuvaa testausprosessin laadukkuudesta kuin jälkikäteen tehtynä ja TMMi-mallin kehittymisen myötä. TMMi-malli oli myös vuonna 2008 vasta julkaistu ja kehitystyön alla ja se on edelleen kehittynyt vuosien aikana ja vastaa nyt pitkälti kokonaisvaltaista testauksen arvioimiseen sovellettavaa mallia (Cannegieter & van Veenendaal 2013, s. 2). Täytyy myös huomioida, että testauksen prosessia arvioiva malli, eli TMMi ei taas

ole prosessin kehittämiseen tarkoitettu malli ja näin myös TMMi:n malli vaatii periaatteessa rinnalle prosessia kehittävän mallin. Altéa-projektissa teimme projektin aikana parannuksia testauksen kehittämiseen tietämättämme, että näillä on vaikutuksia myös testauksen laadulliseen parantamiseen. Jälkeenpäin arvioituna toimintamme testauksen kehittämisen osalta oli erittäin arvokasta ja paransi paljolti projektin testauksen laatua, hallinnointia ja arviointia aikataulun suhteen. Myös esittämäni arviot testausprosessin parantamisessa täyttyivät, kuten esitin tämän luvun kohdissa 5.1-5.3. TMMi:n kypsyystason viisi, eli kehittämisen prosessialueita en ole arvioinut, koska näiden prosessin osa-alueiden saavutettavuus yrityksessä, missä ohjelmistokehitys tai testaus ei ole osa ydinliiketoimintaa ei sinänsä ole mielekästä. Lisäksi kehitimme jatkuvasti testausprosessia koko Altéa-projekin ajan, mutta nähdessämme kehitettyjen testauskäytäntöjen kyvykkyyden, oli myös hyväksyttävä sen vastaavan tarpeitamme jo tasolla kolme ja neljä. Myös tasolta neljä, eli mittaamisen osa-alueesta jäi joidenkin osa-alueiden tiettyjä kokonaisuuksia täyttämättä, joten tason viisi arviointia ei ole näin edes järkevää suorittaa.

6 JOHTOPÄÄTÖKSET

Projektin tavoitteena oli ensisijaisesti suorittaa onnistuneesti Altéa-ohjelmistojen käyttöönotto ja uusia lentokoneen kääntöä tukeva prosessi. Projektin aikana kehitettiin myös Finnairin maapalvelun liiketoimintaprosesseja uusien Altéa-ohjelmistojen tarjoamista lähtökohdista. Tämän nojalla projektissa suoritettiin mittava käyttötapausten tunnistaminen liiketoiminnan kannalta oleellisten asioiden osalta ja testattiin käyttötapaukset testitapausten avulla. Lisäksi tehtiin ohjelmiston perustestaamista ja tarjottiin merkittävissä määrin koulutuspalveluista kaikille prosessiin jollakin tavalla osallistuville tahoille, jotta käyttöönotto olisi ylipäänsä mahdollista. Testauksen ohjaamiseen ja suunnitteluun sekä laadun kehittämiseen tällä osa-alueella tarvittiin testausprosessiin ja johtamiseen liittyvää osaamista ja malleja. Näiden avulla voitiin kehittää Finnairin lähtökohdilta testaukseen toimintatapa, jonka avulla vastaavasti voitaisiin tulevaisuudessa testata uusia ohjelmistoja ja Altéa-tuotteiden tapauksessa suorittaa uusien ohjelmistoversioiden regressiotestaukset, ohjata, valvoja ja johtaa tätä toimintaa. TMMi toimi tässä osaltaan arvioimisen työkaluna testausprosessin kehittymisen tason mittauksessa.

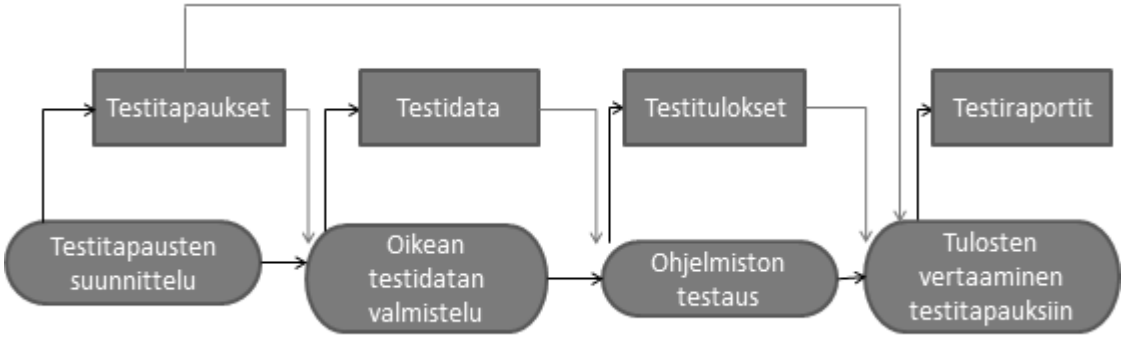
6.1 Testauksen ja testausprosessin opit

Testauksen prosessin kehittämisen lähtökohdat olivat otolliset, koska aluksi ei ollut minkäänlaista suunniteltua testauksen kehittämiseen valittua lähestymistapaa. Ensimmäisenä tehtävänä oli tutustua ohjelmiston määritysdokumentaatioon, ohjelmistoon ja saada koulutus kouluttajalta ohjelmiston käyttöä varten. Ymmärtää eri moduulien käyttötarkoitukset ja näissä tehtävien muutosten syy – seuraus -suhteet. Lisäksi ammatillisen osaamisen taustan perusteella oli tärkeää tuoda oman erikoisosaamisen mahdolliset rajat tiedoksi muulle projektiryhmälle. Tältä pohjalta voitiin kehittää testauksen läpiviemistä ja testausprosessia pienin askelin.

Testaukseen liittyen kokonaisuuteen kuului selkeästi varsinaisen testauksen suunnittelu, eli varsinainen toimintapa, dokumentointi testaukseen liittyen ja prosessi tähän. Toisen osa-projektien alkaessa oli jo muodostunut selkeä käytäntö, että miten käyttötapaukset kuvataan ja dokumentoidaan, millaiset testidokumentit moduuleittain toteutetaan ja miten ne katselmoidaan. Oli myös selkeä tapa kirjoittaa testitapaukset auki ja dokumentoida testauksen osalta syntynet testitulokset ja virheiden raportointikäytäntö oli myös asetettu projektin tarpeisiin. Lisäksi tässä vaiheessa myös testauksen hallinta ja sen raportointi oli kehittynyt siinä määrin, että kaikista moduulien testausdokumentaatioista pystyttiin hakemaan automaattisesti kunkin vaiheen

eteneminen raportoinnin koontiexceliin (katso esimerkki liitteestä 6). Näin viikoittainen seuranta takasi projektin johdolle tarvittavat tiedot testauksen etenemisestä ja mahdollisista ongelmakohteista testaukseen liittyen. Testauksen etenemisen esti joidenkin toimintojen osalta avatut virheraportit, joiden korjaaminen kesti kauan, jopa 3-6 viikkoa. Myös liian myöhään havaitut virheet toiminnallisuuksista ja näistä avatut virheraportit, estivät ohjelmiston käyttöönottoa. Tältä pohjalta myös jokaisen muutoshallinnan kautta tilatun muutostyön testaaminen annettiin suoraan jonkin järjestelmäasiantuntijan vastuulle.

Testaukseen liittyvät raportointikäytännöt olivat kehittyneet viikoittaisiksi ja viikkotasolla niiden vastuut oli jaettu projektin kesken. Näille oli asetettu aikataulut viikkotasolla ja myös varamiesjärjestely ja eskalointiprosessit olivat selkeitä, jos testaus ei edistynyt suunnitellusti. Projektissamme testausprosessin periaate noudatti lähes suoraan alla olevaa kuvaa 6.1:



Kuva 6.1. *Testauksen läpivienti testitapausten avulla (Sommerville 200, s. 539).*

TMMi:n testauksen kypsyystasoihin verrattaessa voidaan arvioni osalta todeta, että päädyimme TMMi:n tasolla kolme: toimintatavat ja tasolta neljä: mittaaminen saavutimme osan kokonaisuuksista, mutta viitostasoa tässä testausprosessin kypsyydessä ei ole vielä saavutettu. Altéa-projektissa toteutui selkeästi hallinnan laadukkuus, eli perusdokumentaation tuottaminen, testausympäristöt olivat kunnossa ja testauksen edistymistä seurattiin sekä hallinnoitiin. Toimintatavat TMMi:n tasolla kolme olivat myös kunnossa. Heikoimpana TMMi:n tasolla kolme toteutui katselmointi, mitä varten projektissa ei järjestetty varsinaisia ohjelmiston katselmointitilaisuuksia koko projektin henkilöstölle Altéa FM -projektin aikana. Myös TMMi:n tasolla neljä on vielä kehitettävää ja standardisoitavaa, jotta prosessin kehittäminen toisi testauksen hallintaan automaatiota. Mutta joltain osin myös TMMi:n kypsyystaso neljä kuitenkin toteutui, kuten esimerkiksi testauksen mittaaminen ja myös joltain osin ohjelmiston laadun mittaaminen Finnairin vaatimuksia vasten. Täytyy myös muistaa, että prosessin kehittäminen Altéa-projektissa oli jatkuvaa ja kokonaislaatuun tähdättiin Finnairin vaatimusten mukaisesti. Mutta koska kaikkia tason neljä osa-kokonaisuuksia saatu oman arvioni mukaisesti saavutettua niin viidennen kypsyystason arviointia ei ole syytä lähteä edes tarkemmin suorittamaan. Suoritustaso on kuitenkin tarpeeksi hyvä

organisaatiolle, jonka lähtökohdat testaamiselle olivat lähes nollatasolla ja jonka päätoimialueeseen ohjelmistojen testaus ei kuulu. Testausprosessin suunnittelu ja erityisesti sen vertailu olemassa olevaan dokumentaatioon ja TMMi-malliin ja muihin malleihin tutustuminen oli itselleni tärkeässä osassa projektia ja diplomityötä, koska näin sain vertailukohtaa tähän työhön.

6.2 Kehittämisen edut projektissa ja testausprosessin parantaminen

Testauksen lähtökohdat ja tavoitteet on tärkeää määritellä kussakin tapauksessa. Erityisesti tärkeää on myös määrittää pisteet, jolloin testaus on täyttänyt tehtävänsä ja on saavutettu suunniteltu lopputulos testauksen osalta ja pystytty laadunvarmistukseen vaatimusten mukaisesti. Näin testaukselle on mahdollista kuvata testauksen strategia ja testauskäytännöt. Finnairilla testaus keskittyi lähinnä liiketoiminnalle kriittisten toiminnallisuuksien testaamiseen ohjelmistomoduuli kerrallaan järjestelmätestauksen osa-alueella. Merkittävää on ymmärtää, milloin on testattu tarpeeksi ja milloin testaamisessa tai projektissa on syytä siirtyä seuraavaan vaiheeseen. Myös liiketoiminnan kannalta merkittävät ja tärkeät ominaisuudet, sekä toiminnallisuudet, on syytä saada mukaan ohjelmistoon, jos näin on liiketoimintaprosesseissa määritelty. Altéa-projektissa haasteelliseksi osoittautui muutoshallinnan kautta tilatut Finnairin ja eurooppalaisille lentoyhtiöille tärkeät ominaisuudet, joiden testaamiseen projektissa kiinnitettiin erityistä huomiota.

Alihankkijan roolia ei voi vähätellä ja heidän kanssa hoidettiin perustestaus Intiassa ja ei-toiminnallinen testaus sekä myöhemmin uusien ohjelmistoversioiden julkaisun yhteydessä regressiotestaus. Alihankkijan rooli ei jäänyt vain tähän, vaan heiltä saatiin myös apua testaamiseen dokumentointiin ja raportointiin, lisäksi tällä yrityksellä oli valmiina tietyn tyyppinen prosessikehitys ja heillä on CMMI:n mukainen tunnistettu prosessien kehittämisen kypsyys tasolla viisi (5), joka on korkein, mikä voidaan saavuttaa**.** CMMI:n kypsyystason viisi saavuttanut yritys pystyy keskittymään prosessien kehittämiseen, koska heidän toiminnassaan tämä on keskiössä kaikessa toiminnassa. Alihankkijalla oli myös oiva mahdollisuus tutustua uuteen ohjelmisto-osa-alueen kehittämiseen. Finnair taas sai yritykseltä myös testauksen hallintaan apu ja sen lisäksi standardisoitu ja jatkuva kehittäminen ITC Infotechin osalta toi projektiin arvokasta osaamista mahdollistamalla Finnairin henkilöstölle uusien asioiden oppimisen.

Yhteistyökumppaneiden rooli korostui erityisesti eri sovellusten viestiyhteyksien ja ohjelmistointegraatioiden testaamisen osalta, joiden ”omistajina” oli jokin muu osasto tai taho kuin varsinaisesti Finnair tai Finnairin kaupallinen ryhmä. Erityinen suunnittelu testauksen ja sen läpiviemiseksi eri työvaiheineen oli avainasemassa testauksen onnistumiseksi ohjelmistointegraatioiden osalta. Integraatioiden testaamiseen liittyi muiden järjestelmien koulutus, ohjelmistojen käyttö ja niiden testiympäristöihin

tutustuminen, testaaminen Amadeuksen, eli toimittajan kanssa ja mahdollisia muita työvaiheita. Ohjelmistointegraatioiden suunnittelu ja tietojen siirtymisen validointi oli tärkeässä osassa toimivan systeemin ja ohjelmistokokonaisuuden saavuttamiseksi, jotta tiedon tuottajan tietoa voitiin käyttää suoraan hyväksi Altéa-ohjelmistoissa. Ilman tiettyjä ohjelmistointegraatioiden toteuttamista ja testaamista Altéa FM -ohjelmistossa ei olisi ollut mahdollista ottaa ohjelmistoa tuotantokäyttöön.

Dokumentaation merkitys oli kiistämätön ja ennen kaikkea tuotannossa järjestelmäasiantuntijoiden kokemus oli ensiarvoisen tärkeää. Testitapausten toteuttamisessa yhdistettiin liiketoimintaosaaminen ja ohjelmiston määrityksiä vastaan kohdistuva testaaminen. Näin saatiin testidokumentaatio, joka vastasi sekä Finnairin tarpeita että testasi ohjelmiston määrityksiä vasten ohjelmiston toimintaa. Dokumentaation merkitys koko projektin onnistumiselle oli selkeä ja sen takia oli myös tärkeää, että dokumentaatio oli saman mallin mukaisesti laadittu ja siinä myös raportoitu testauksen tulokset yhteisesti sovitun mallin mukaisesti. Lisäksi dokumentaation tarve jatkotestausta varten oli kiistämätön, koska silloin testitapaukset voitiin toteuttaa jo ennalta dokumentoidun testitapauksen mukaisesti ja tarvittaessa olla yhteydessä aikaisemmin kunkin testitapauksen suorittaneeseen henkilöön. Regressiotestaus ja uusien ohjelmistoversioiden testaaminen oli myös mahdollista helpohkosti, koska dokumentaatio oli sovitun mallin mukaisesti toteutettu ja se löytyi ennalta sovitusta paikasta, jolloin sen hyödyntäminen oli kaikille testaukseen osallistuville mahdollista.

Testausdokumentaatio muodosti projektin aikana projektin oman standardidokumentaation ja tämä soveltui testausprosessin pohjadokumentaation. Koska dokumentaatio tuotettiin testauksen osalta moduulikohtaisesti ja moduuleita oli ohjelmassa useampi, vaadittiin myös testauksen hallintaa. Testauksen hallinta tarkoittaa dokumenttien versiointia, tulosten raportointia moduulikohtaisesti, ohjelmistointegraatioiden osalta ja koko ohjelmiston testauksen edistymisen raportointia. Testauksen hallinta ja raportointi toi projektiin uusia vaatimuksia, eli raportoinnin kehittämistä ja muutoshallinnan käytäntöjen kuvaamista testaukseen liittyen. Muutoshallintaan liittyivät testauksen kannalta uuden toiminnallisuuden testaaminen ja toiminnallisuuden testauksen raportointi. Pohjana tässä käytettiin käyttötapauksia, jotka kirjoitettiin auki yhdessä toimittajan kanssa. Muutoshallintaan hyväksytyille ominaisuudelle kirjoitettiin testitapaukset, joiden avulla nämä myöhemmin testattiin toiminnallisuuden toimivuuden osalta. Tätä toimintatapaa voidaan pitää tietyltä osalta laadun varmistuksena ja myös katselmointina, jotta ohjelmiston muutoshallinnan dokumentaation pohjalta saadaan ensimmäisellä kerralla toteutettua mahdollisimman tarkasti vaatimuksia vastaavat toiminnallisuudet.

Kehittämisestä on aina yleensä etua, erityisesti, jos se tehostaa toimintaa ja helpottaa asioiden hallinnointia, kuten Altéa-projektissa. Valmiita malleja tai standardeja on hyvä käyttää hyödyksi, jotta kaikkea ei tarvitse keksiä itse ja mallien avulla myös kehittäminen helpottuu, koska käytössä on jokin jo toimivaksi todettu formaatti. Sommervillen, Hetzelin ja CMMI:n opit ja teoriat olivat tiedossa ja mahdollisuuksien

mukaan käytössä. Tunnettujen teorioiden ja mallien käyttäminen helpottaa ja nopeuttaa suunnittelutyötä ja siten myös antaa suuntaviivan toiminnan kehittämiselle.

6.3 Loppuyhteenveto

Testauksen avulla on mahdollista todentaa testattavan kohteen toimivuus, eli ohjelmistosovelluksen tai jonkin muun tuotteen toiminnalliset ja ei-toiminnalliset ominaisuudet ja osoittaa näissä ensinnä määrittelyyn nähden liittyvät puutteet tai sitten osoittaa muulla tavoin, että ohjelmiston toiminnallisuuksissa on virhe. Jos ohjelmiston suunnittelussa ja määrittelyvaiheessa on otettu huomioon vain tietyn tyyppinen kokonaisuus ja siihen määritellyt käyttötapaukset, ollaan tilanteen edessä, jossa asiakas tai tilaaja joutuu muutoshallinnan kautta tilaamaan ohjelmistoon lisäominaisuuksia. Näitä muutoskohteita löytyi myös Altéa-projektissa, joista ehkä merkittävimpänä ja laajimpana muutoksena oli vapaan istumajärjestyksen hallinta ohjelmiston toiminnallisuuksien osalta.

Testauksen suunnittelu, dokumentointi ja hallinta ovat avainasemassa testauksen onnistumisen kannalta. Tähän liittyy tietenkin projektissa myös aikataulu, edistymisen raportointi ja edelleen esimerkiksi resurssien kohdentaminen oikein. Altéa-projektin laajuisessa projektissa testausprosessin ja sen kehittämisen näkökulma oli tärkeä osa projektia, eli projektissa oli mahdollista kehittää testausta, ja parantaa sitä tarpeen mukaan. Testaus kokonaisuutena nähtiin tärkeäksi testaamisen käytäntöjen kehittämiseksi ja näin saatiin myös yhteisiä mallien ja toimintatapoja projektille. Tätä tukivat kauemmin projektissa mukana olleet henkilöt testauksen toimintatapojen käyttöönotossa projektin uusille resursseille. Näin kaikilla projektiin osallistuneille oli tavoitteena taata yhtäläiset lähtökohdat dokumentaation tuottamiseen ja testauksen läpivientiin. Projektin sisäinen tapa dokumentoida ja toteuttaa dokumentteja osaltaan helpottaa koko testauksen toteuttamista ja seurantaa, koska kaikilla tuotetuilla dokumenteilla on määrämuotoinen malli, jonka pohjalta ne aina toteutetaan. Erityisesti testauksen raportointi kehittyi projektin aikana palavereissa epämuodollisesti tavasta jakaa tietoa kehittyneeksi Excel-raportoinniksi, jonka avulla voitiin ennustaa testitapausten suorittamisen päätepiste ajan funktiona. Raportointityökalu takasi tarkan tiedon kaikkien moduulien testauksen edistymisen osalta. Raportista selvisi suoritetut testitapaukset suhteessa suorittamattomiin, aukiolevien virheiden määrä, alustava testausaikataulu moduulia kohden ja arviot muutoshallinnassa olevien testitapausten suorittamisesta. Raportointikäytäntöjen kehittäminen takasi myös sen, että viimeisin tieto oli saatavilla tarvittaessa nopeastikin kootun raportin pohjalta. Raportoinnin hallinta oli aluksi haastavaa, koska raportin keräämiseen liittyi usean raportin koostaminen yhdeksi kokonaisuudeksi. Raportointi kuitenkin kehittyi myös projektin aikana, jonka tuloksena testauksen kokonaisseurantaan liittyviä raportteja saatiin koottua taulukko-ohjelmiston linkityksien avulla nopeasti.

Testauksen prosessien kehittämisen suurimpia haasteita oli se, että testaaminen ei ole koko organisaation päätoimintaa, ei edes Finnairin IT-osastolla tai kaupallisessa ryhmässä. Finnairilla on tärkeää tuntea testaukseen liittyvä prosessi ja sen käyttö sovelluksien verifioinnissa, mutta varsinaisesti ohjelmistojen testaaminen ei ole Finnairin ydinliiketoimintaa. Sen takia projektiin osallistuvien sidosryhmien sitominen kestävään testauksen hallintaan ja prosessin kehittämiseen oli haastavaa. Altéa-projektissa kuitenkin ymmärrettiin testausprosessin kehittäminen osana projektia, ja näin ollen testauksen hallintaa oli mahdollista kehittää ja tehostaa myös testaamista ja raportointi. Tiedossa oli myös, ettei kaikki ohjelmiston elinkaareen liittyvä testaus tule pysymään Finnairin sisällä ja alustavasti oli sovittu regressiotestauksen toteuttamisesta ulkoisesti.

Altéa-projektissa testaukseen panostamiseen ja sen kehittämiseen vaikuttivat järjestelmäasiantuntijoiden palkkaamiseen projektiin Finnair-konsernin sisältä, mikä johtui erityisesti jokaisen henkilön liiketoimintaosaamisesta. Projektiin osallistuneilla järjestelmäasiantuntijoilla ei ollut itseäni lukuun ottamatta kokemusta tai osaamista ohjelmiston testauksesta ja ohjelmistojen kehittämiseen liittyvistä työvaiheista. Toinen painava syy testaukseen panostamiseen Altéa-projektissa oli se, että ohjelmistojen tekninen toteutus ei Altéa FM -ohjelmiston käyttöönotossa vastannut Finnairille spesifisiä vaatimuksia. Altéa CM -ohjelmiston käyttöönotossa testaus oli liiketoiminnan kannalta erittäin tärkeässä roolissa, sillä Finnairin käyttöönotto oli koko ohjelmiston ensimmäinen käyttöönotto koko ohjelmistolle. Vaikka projekti tuntui toisinaan keskittyvän yhteen painopistealueeseen kerrallaan, eli esimerkiksi testaukseen tai koulutukseen, niin testauksen yhteisten käytäntöjen edistäminen ja kehittäminen projektin aikana olivat avainasemassa testauksen laadun parantamiseksi. Tämä näkyi esimeriksi muutoshallinnan jälkeen toteutettavissa toimissa ja toisessa projektissa Altéa CM -ohjelmiston testauksen kokonaisuudessa. Testausta projektin aikana tukivat sekä oma organisaatio, toimittaja, eli Amadeus ja sen lisäksi ulkoa hankittu tietotaito. Näin pystyttiin vähentämään koko projektiin liittyviä riskejä, kun koko projektin aikana olivat tietyt suuntaviivat testaamisen suorittamiseksi, ja tukea oli tarvittaessa saatavilla toimittajalta, sisäisesti ja alihankkijalta.

Testauksen kannalta jälkikäteen pohdittaessa testausprosessi kehittyi niihin tarpeisiin, jotka projektissa olivat tarpeen sen läpiviemiseksi. Testauksen kehittäminen kasvoi organisaatiossa TMMi:n tasolta yksi merkittävästi sinä aikana kun projekti oli aktiivinen. Avainasemassa oli uuden oppiminen ja vanhoista tavoista myös poisoppiminen ja muutoksen mahdollistaminen testausta omana kokonaisuutena ajateltaessa. TMMi:n mallin viiden kyvykkyystasoihin peilaten oman arvioni mukaan saavutimme tason kolme. Tasolta neljä saavutimme osan kokonaisuuksista ja tällä kypsyystasolla mittaaminen ja laaduntarkkailu ovat tärkeässä roolissa. Näiden osa-alueiden osalta olisi voinut asioita ja toimintatapoja edelleen kehittää, mutta Altéa-projektissa mittaaminen ja laaduntarkkailu tuotiin Finnairin vaatimuksia vastaavalle tasolle. Arvioinnin osalta täytyy myös muistuttaa, että tulos perustuu omaan arviooni teoriaan ja arviointikäytäntöjen soveltamiseen pohjautuen. Prosessin kehittämisen

kannalta tämä on olennaista, eli parantaa olemassa olevaa tapaa toimia ja tehostaa sitä, sekä dokumentoida toimintatavat jälleenkäytettäväksi. Testausprosessin kehitys tapahtui Altéa-projektissa muiden projektiaktiviteettien ohella ja Finnairille jäi kokemusta testauksen prosessin kehittämisestä ja ennen kaikkea ohjelmistotestauksen läpiviemisestä ohjelmistoprojektissa ja käyttöönottoprojektissa. Kehitettyä toimintatapaa ja mallidokumentaatiota olisi helposti mahdollista käyttää tulevaisuudessa muiden ohjelmistoprojektien testauksessa hyödyksi. Testauksen kehittäminen tuotti Finnairille prosessin, jossa kuvattiin käyttötapaukset ja näiden testitapaukset, ohjattiin testauksen edistämistä, hallinnoitiin testauksen dokumentaatiota ja raportoitiin testitukoksia. Tämän lisäksi testausta kehitettiin toteuttamalla dokumenttipohjia, joiden avulla testauksen dokumentointi ja raportointi oli tehokkaasti mahdollista. Testauksen kehittäminen oli erittäin tärkeässä roolissa, jotta projektin hallittavuus oli ylipäänsä mahdollista. Raportoinnin avulla voitiin kohdentaa testaukseen käytettyjä resursseja ja ennustaa esimerkiksi eri moduulien osalta testauksen valmistumista näiden osalta.

Jatkokehittämisen osalta on mahdollista suorittaa virallinen TMMi:n arviointi tarvittaessa tai johonkin testauksen arviointimalliin pohjautuen. Tästä tietenkin seuraa jonkin verran kustannuksia, mutta näin toimittaessa testausprosessit on arvioituna ja malli täysin dokumentoitu. Näin saadaan myös tarvittavat kehityskohteet ja parantaminen testauksen osalta aikataulutettua. Tietenkin ennen arviointia on hyvä pyrkiä saavuttamaan TMMi:n kypsyystason neljä osalta tarvittavat alikokonaisuudet, kuin ne ovat Finnairille merkityksellisiä ja määritellä nämä osaksi jatkuvaa testausprosessia. Lisäksi on mahdollisesti pyrkiä TMMi:n tasolle viisi, jos tämä nähdään tarpeelliseksi. TMMi:n tasolta viisi voidaan myös tunnistaa sellaisia kokonaisuuksia, joiden panostaminen on syntyviin kustannuksiin nähden järkeviä, mutta eivät kokonaistoiminnan tai -kehittämisen kannata turhia. Testausprosessin liittyvät asiat, dokumentaation laajuus, toimenpiteet ja muut kehittämiseen liittyvät asiat on hyvä dokumentoida tavalla tai toisella ja esimerkiksi kirjata nämä esimerkiksi laatusuunnitelmaan ja tuoda nämä testaukseen liittyvät kokonaisuudet mahdollisesti osaksi koko organisaation toimintaa. Jatkuvan kehittämisen avulla testauksen osalta voidaan pitää yllä sellaista ammattitaitoa, jonka tietotaidon käyttäminen muissa vastaavissa projekteissa on varmasti arvokasta ja lisäksi osaamista voidaan jakaa esimerkiksi muiden lentoyhtiöiden kanssa. Tämä voi taata tiiviin yhteistyön ja mahdollisuudet uusien kokonaisuuksien osalta myös laajemmin Finnairin liiketoiminnassa.

LÄHTEET

Arovaara, J. 2008. Testausprosessin kehittäminen – keskeisiä kysymyksiä. Jyväskylän yliopisto. Jyväskylä. 120 s. [WWW]. [Viitattu 12.5.2013]. Saatavissa: https://jyx.jyu.fi/dspace/bitstream/handle/123456789/18883/URN_NBN_fi_jyu-200808265679.pdf?sequence=1.

Bach, J., Kaner, C. & Pettichord, B. 2002. Lessons Learned in Software Testing. A Context-Driven Approach. New York. Wiley Computer Publishing. John Wiley & Sons, Inc. 286 s.

Boehm, B. W. 1987. Improving Software Productivity. IEEE Computer. Nro 9. S. 43-57.

Cannegieter, J. & Van Veenendaal, E. 2013. Test Maturity Model Integration (TMMi): Results of the first TMMi benchmark - where are we today? EuroStar Software Testing Community. 10 s. [WWW]. [Viitattu 4.6.2013]. Saatavissa: http://www.tmmi.org/pdf/e-book_TMMi.pdf.

Drabick, R., D. 2004. Best Practices for the Formal Software Testing Process. New York. Dorset House Publishing. 286 s.

Evans, K. Testing in Scrum Projects. 2008. [WWW]. [Viitattu 5.5.2013]. Saatavissa: http://www.cs.tut.fi/tapahtumat/testaus08/Kalevi.pdf.

Gens, W. 2011. End2End Testing: It‘s Just the Beginning. TE Testing Experience, The Magazine for Professional Testers, Nro 14. ISNN1866-5705. S. 68-69. [WWW]. [Viitattu 12.5.2013]. Saatavissa: http://testingexperience.com/testingexperience14_06_11.pdf.

Goslin, A. 2009. TMMi Assessment Method Application Requirements (TAMAR), Version 2.0. TMMi Foundation. 26 s. [WWW]. [Viitattu 9.6.2013]. Saatavissa: http://www.tmmi.org/pdf/TMMi.TAMAR.pdf.

Haikala, I., & Märijärvi, J. 2002. Ohjelmistotuotanto. 8. uudistettu painos. Pieksämäki/Helsinki (julk.). Talentum Media Oy. 430 s.

Hetzel, B. 1998. The Complete Guide to Software Testing. Second Edition. New York. A Wiley-QED Publication. John Wiley & Sons, Inc. 284 s.

Immonen, J. 2002. Johdatus ohjelmistotuotantoon. Joensuun yliopisto. [WWW]. [Viitattu 8.4.2012]. Saatavissa: http://cs.joensuu.fi/~jimmonen/jot_moniste/jot_moniste_121.html.

Jääskeläinen, A., Katara, M. & Vuori, M. Ohjelmistojen testaus, 2013 -luentokalvot. Tampereen teknillinen yliopisto. 525 s. [WWW]. [Viitattu 5.5.2013]. Saatavissa: http://www.cs.tut.fi/~testaus/s2012/luennot/OHJ-3060_2012.pdf.

Kolehmainen, A. 2012. Tietoviikko. STT: It-vika pysäytti Finnairin lennot. [WWW]. [Viitattu 10.3.2012]. Saatavissa: http://www.tietoviikko.fi/cio/stt+itvika+pysaytti+finnairin+lennot/a766679?s=l&wtm=tietoviikko/-30012012&.

Kollanus, S. 2006. Testauksen erityisalueita. 2 s. [WWW]. [Viitattu 10.3.2013]. Saatavissa: http://users.jyu.fi/~kolli/testaus2006/materiaali/testauksen_erityisalueita.pdf.

Koomen, T. & Pol, M. 1999. Test Process Improvement. A practical step-by-step guide to structured testing. Reading/Edindburgh (julk.). Pearson Education Limited. 218 s.

Roodenrijs, E. 2009. End-to-end Testing. [WWW]. [Viitattu 10.5.2013]. Saatavissa: http://www.testingthefuture.net/tag/end-to-end-testing/.

Sommerville, I. 2007. Software Engineering 8. Edinburgh. Addison-Wesley Publishers Limited ja Pearson Education Limited. 840 s.

Stenberg, A. 2007a. Ohjelmiston testaus 2007 – johdanto -kalvosarja (53 kalvoa). Tampereen teknillinen korkeakoulu, Porin yksikkö. 27 s. [WWW]. [Viitattu 10.3.2013]. Saatavissa: http://www.pori.tut.fi/~stenberg/index_files/johdanto.pdf.

Stenberg, A. 2007b. Ohjelmiston testaus 2007 – testausprosessi -kalvosarja (17 kalvoa). Tampereen teknillinen korkeakoulu, Porin yksikkö. 9 s. [WWW]. [Viitattu 10.3.2013]. Saatavissa: http://www.pori.tut.fi/~stenberg/index_files/Page625.htm.

Stenberg, A. 2007c. Ohjelmiston testaus 2007 – testauksen hallinta -kalvosarja (59 kalvoa). Tampereen teknillinen korkeakoulu, Porin yksikkö. 30 s. [WWW]. [Viitattu 10.3.2013]. Saatavissa: http://www.pori.tut.fi/~stenberg/index_files/hallinta.pdf.

Szalvay, L. 2009. Scrum Methodology. [WWW]. [Viitattu 5.5.2013]. Saatavissa: http://scrummethodology.com/.

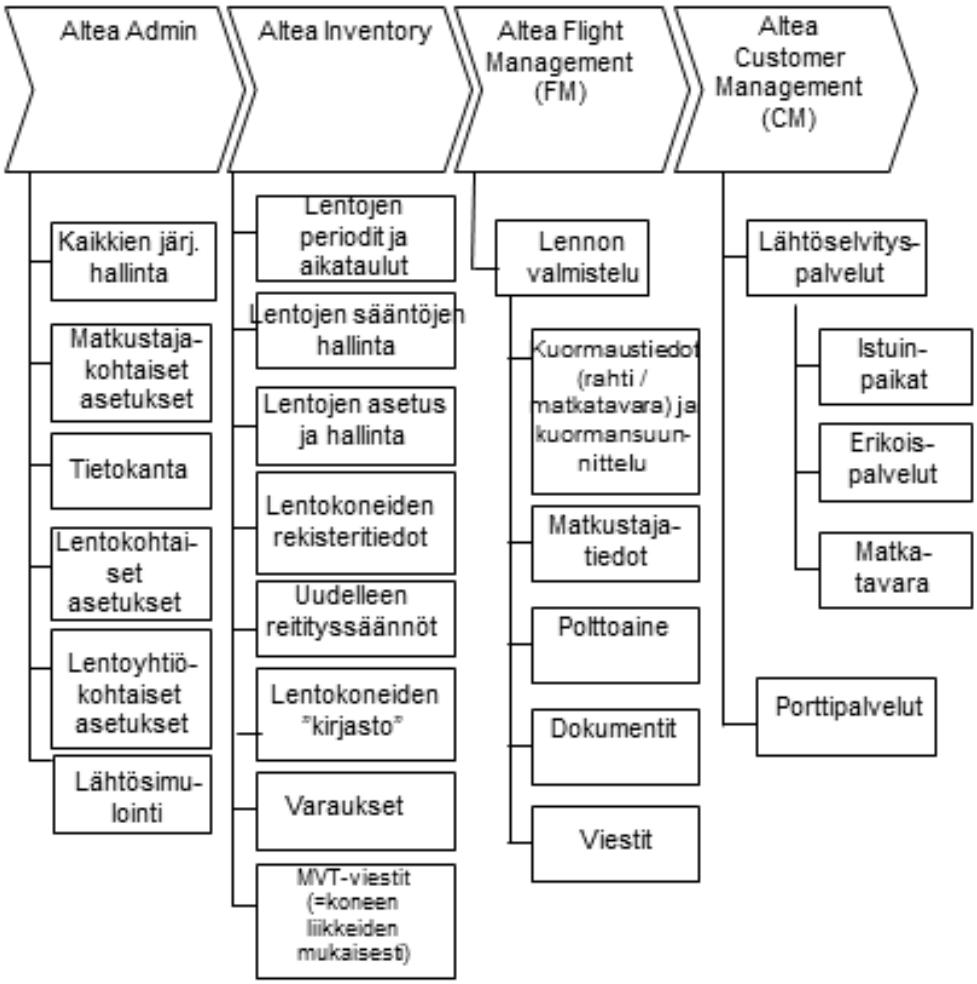
Taina, J. 2007. Ohjelmistojen testaus - luentokalvot; ohte-2_6.pdf. Helsingin yliopisto. 6 s. [WWW]. [Viitattu 12.3.2013]. Saatavissa: http://www.cs.helsinki.fi/u/taina/ohte/s-2008/luennot/ohte-2_6.pdf.

Tersa, T. 2002. Testausmenetelmien käyttö sovelluksen systeemitestausvaiheessa. Tietotekniikan Pro Gradu -tutkielma. Jyväskylän yliopisto, tietotekniikan laitos. 139 s. [WWW]. [Viitattu 30.3.2012]. Saatavissa: http://www.mit.jyu.fi/opetus/opinnayte/gradu/systeemitestaus/Tiina_Tersa.pdf.

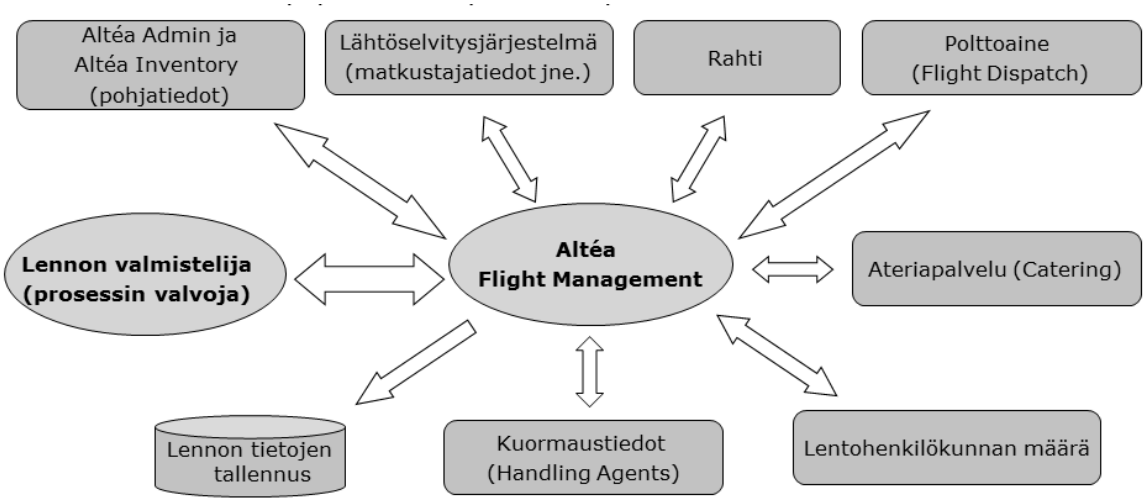
Tuomikoski, J. 2009. Testing in Scrum. Oulun yliopisto. 21 s. [WWW]. [Viitattu 5.5.2013]. Saatavissa: http://www.tol.oulu.fi/users/ilkka.tervonen/Ote_vierailu_09.pdf/.

Van Veenendaal, E. 2012. Test Maturity Model Integration (TMMi), Release 1.0. TMMi Foundation, Irlanti. 219 s. [WWW]. [Viitattu 26.3.2013]. Saatavissa: http://www.tmmi.org/pdf/TMMi.Framework.pdf.

LIITE 1: ALTÉA-OHJELMISTOPERHEEN OHJELMISTOT JA NIIHIN LIITTYVÄT AVAINTOIMINNALLISUUDET



LIITE 2: KAAVIO, JOSSA ON KUVATTU YLÄTASOLLA ALTÉA FM -OHJELMISTOON TULEVAT JA SIITÄ LÄHTEVÄT TIETOVIRRAT ERI TOIMIJOILTA



LIITE 3: FINNAIRIN ALTÉA-PROJEKTIN HENKILÖSTÖ

Projektiryhmä

Matti Alanne
Change Management
Finnair Impacted Applications

Michael Ekström
Administration and Security

Veikko Vuorela
FM Project Manager

Elina Grönlund
IATCI
Test Management
Administration and Security

Sirpa Vyyryläinen
eTicket
Self Service

Marita Kolehmainen
Test Coordination

Tero Lehtoranta
FM System Business Support

Matti Lehtonen
Finnair Impacted Applications
IT Support

Kari Pauro
Project Director

Joel Vuolle-Apiala
Business Support
CLC Processes

Tuija Makkonen
Self Service

Pia Pakalén
Training Manager
Project Coordination

Jouni Suvilinna
Installations and Devices

Maarit Sylvander
CM Project Manager

Heidi Uotila
Testing

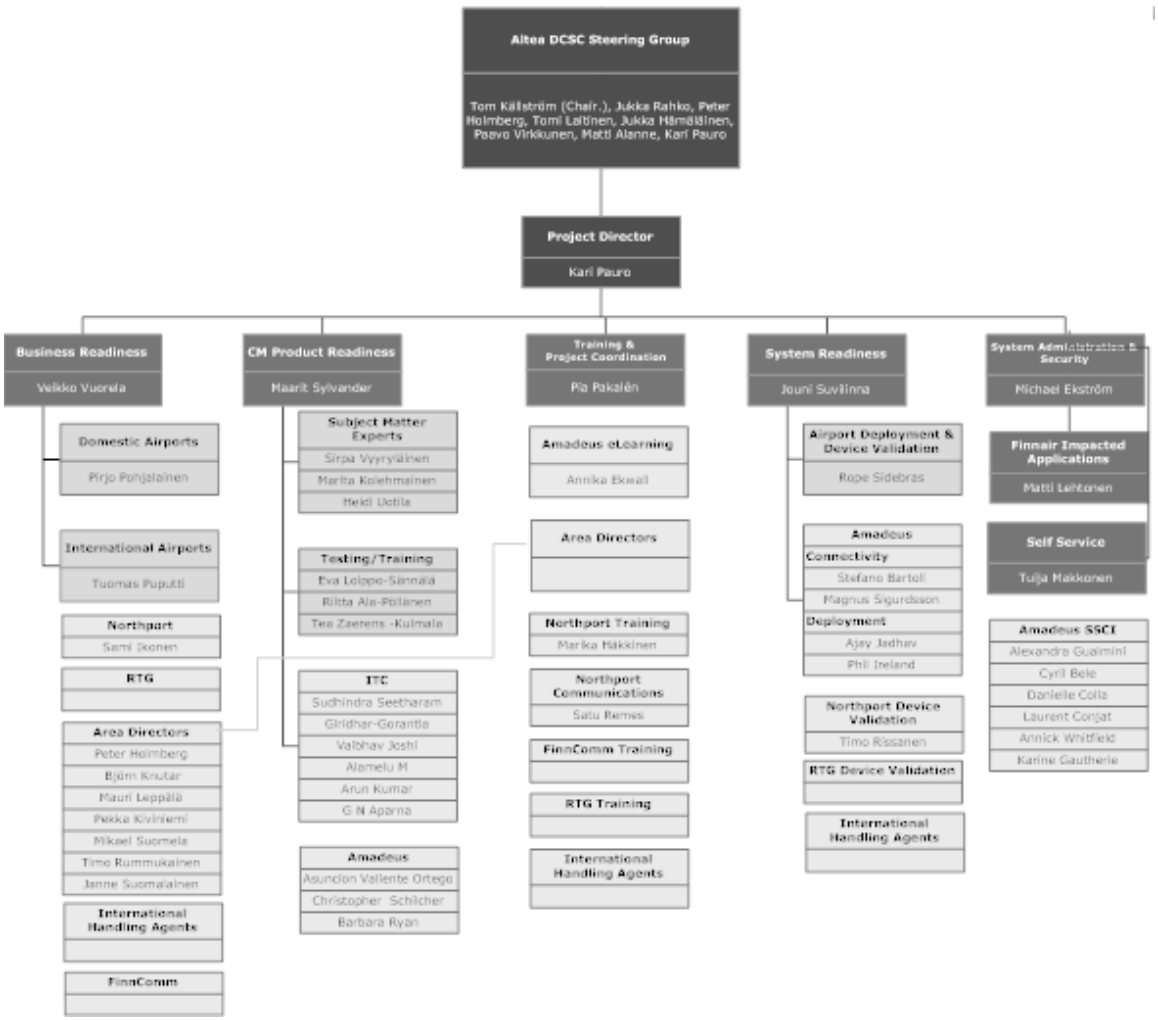
Pirjo Pohjalainen
Testing/Training
CM Implementation

Tuomas Puputti
Testing/Training
CM Implementation

Tuula Ihamäki
Testing/Training
FM Support

Rope Sidebras
Testing/Training
CM Device Definitions

LIITE 4: ALTÉA CM -PROJEKTIN ORGANISAATIOKAAVIO ENGLANNIKSI



LIITE 5: MALLI TESTAUSTAULUKOSTA, JOHON TESTITAPAUKSET JA TESTITULOKSET DOKUMENTOITIIN KUNKIN OHJELMISTOMODUULIN OSALTA

Project Name	Finnair DCS - Flight Management
Project id	FIN-DCS-FM
Document Name	Flight Management - Test Case Document
Document Number	
Version	0.2
Date	27-June-2007
Author	Sidebras RK

Controlled

For Distribution List please refer to the Issue Register

Document Approval Records				
Sl. No.	**Approver's Name**	**Designation**	**Signature**	**Date**

Revision History					
Version. Revision	**Date**	**Author / Modified By**	**Reviewed By**	**Module**	**Brief Description of the changes**
0,1	21-06-2007	Sidebras RK	AVN	JFE Test Cases	FM JFE Test Cases - First Draft
0.2	27-06-2007	Sidebras RK	AVN	JFE Test Cases	FM JFE Test Cases - Completed, two test cases tested and sent document for review.

Table of Contents

I. JFE Test Cases		Documents Referred
1	Changing of the Modules from menu	.doc
2	Buttons and response time	
3	Status changes	
4	Use of Ramp module without mouse	
5	Use other modules without mouse (define later which one)	
6	Tests for Fuel as a JFE related testing	.doc
7	Tests for SWA as a JFE related testing	
8	Drop down menus	
9	SI Text	
10	Several rows of commodities added	
11	Background color of the Finnair aircraft	
12	Mandatory and non mandatory fields	
13	Moving load between containers, bulk and offload	
14	Messages and messaging	
15	Triggering flight actions	
16	Proper TAB functioning	
17	Radio buttons and check boxes	
18	Text fields	

19	Screen validation test cases
20	Help screen is displayed
21	Simulation Mode
22	Pop-up windows/Panel testing with some panels
REVIEW COMMENTS	

REVIEW COMMENTS (If any)							
UAT Area/ Subarea	**Test Case Id**	**Step No (if any)**	**Review Comments**	**Remarks of the Reviewer (if any)**	**Accepted/ Rejected (Offshore)**	**Remarks of the Reviewer (if any)**	**Comments (Offshore)**
JFE Test Cases	**1**						
	2						
	3						
	4						
	5						
	6						
	7						
	8						
	9						
	10						
	11						
	12						
	13						
	14						
	15						
	16						
	17						
	18						
	19						
	20						
	21						
	22						

Altéa FM UAT_Test Cases									
Test Cases					**Test Log**				
Test Case Id	**Use Case**	**Steps**	**Test Case Description**	**Expected Result**	**Tested by**	**Date tested**	**Pass/Fail**	**Error ID**	**Comments**
1	**Changing of the Modules from menu**	1	Open one module (load controller)	Load controller module should open					
		2	Open other module (ramp)	Ramp module should open					
		3	Open some third module (freight)	Freight module should open					
		4	Repeat these three operations	All the modules should open in same state as modules were when last time closed.					
2	**Buttons and response time**	1	Press Apply button	Screen update should be successful					
		2	Press Approve Distribution button	Approve Distribution creation should be successful					
		3	Press Update button	Update creation should be successful					
		4	Press Cancel button	All updates in panel where button is presses should be not created					
		5	Press						
		6	Press						
		7	Press						
3	**Status changes**	1	Press Apply button in Ramp screen	Screen update should be successful, status change appear in Status bar and message appear in message panel					

		2	Press Approve Distribution button in Load Control Load distribution screen	Approve Distribution creation should be successful, status change appear in Status bar and message appear in message panel					
		3	Press						
		4	Press						
		5	Press						
		6	Press						
		7	Press						
4	**Use of Ramp module without mouse**	1	Do all task that will be included in Ramp Clearance without mouse	Everything should work as well without mouse like with mouse	Rope Sidebras	27.6.2007	Passed		
		2	Login		Rope Sidebras	27.6.2007	Not tested		
		3	Find Flight		Rope Sidebras	27.6.2007	Passed		
		4	Add Ramp Agent		Rope Sidebras	27.6.2007	Not passed		
		5	Open Modify and View panel		Rope Sidebras	27.6.2007	Not passed		
		6	Modify Load		Rope Sidebras	27.6.2007	Passed		
		7	Add ULD number		Rope Sidebras	27.6.2007	Passed		
		8	Update ULD details		Rope Sidebras	27.6.2007	Passed		
		9	Edit Modify Load panel figures		Rope Sidebras	27.6.2007	Passed		
		10	Add baggage in ULD in Modify Load panel		Rope Sidebras	27.6.2007	Passed		
		11	remove Baggage from ULD in Modify Load panel		Rope Sidebras	27.6.2007	Passed		

		12	Add baggage in Bulk compartment in Modify Load panel		Rope Sidebras	27.6.2007	Passed		
		13	Press cancel button in Modify Load panel		Rope Sidebras	27.6.2007	Passed		
		14	Press Update button in Modify Load panel		Rope Sidebras	27.6.2007	Passed		
		15	Try to get back on screen with the keyboard		Rope Sidebras	27.6.2007	Passed		
		16	Change position of the containers		Rope Sidebras	27.6.2007	Not passed		
		17	Enter a new row in modify load panel		Rope Sidebras	27.6.2007	Not passed		
		18	Drag baggage to other ULD		Rope Sidebras	27.6.2007	Not passed		
		19	Do ramp clearance		Rope Sidebras	27.6.2007	Not passed		
		20	Press Apply button		Rope Sidebras	27.6.2007	Passed		
		21	Do Baggage Variation		Rope Sidebras	27.6.2007	Not tested		
5	**Use other modules without mouse (define later which one)**	1	Do all task that will be included in other modules without mouse	Everything should work as well without mouse like with mouse if designed					
6	**Tests for Fuel as a**	1	Add different kind of fuel figures in Standard fuel	Fuel figures update					

	JFE related testing	2	Change for non-standard and add different kind of figures. Try also fill tanks with maximum and minimum values here and try to use different fuel combinations between tanks	Non-standard fuel figures can be adjusted and changed					
		3	Change back to standard fuel and test what will happened	This operation should be possible without problems					
		4	Change back to non-standard fuel, test if problems will occur, repeat these tests couple of times per aircraft type	This operation should be possible without problems					
7	**Tests for SWA as a JFE related testing**	1	Add different kind of items and weights manually and from the list if it is provided	SWA items is possible to add					
		2	Add different kind of items in different positions in cabin and in compartments. Try also adding items with maximum and minimum values and try to use different item combinations between compartments or cabin.	SWA items could not be placed in certain places where maximum weight is limited. Not too many SWA items can be added in the same position.					
		3	See if there is right effect for the SWA W&B						
		4	Remove items from list and test if problems will occur, repeat these tests couple of times per aircraft type						

8	**Drop down menus**	1	Test all different drop down menus (Pantry, Crew, Move Load, Documents Type, Cabin Load panel etc.) by opening menu and trying to test if, e.g. in the previous case changes will occur when changing form one mode to other (Fuel: standard to non-standard) or adding standard and non-standard items.	JFE functionalities and drop down menu should work as designed so those should open and give all the details in drop down menu that are inserted under that topic.					
		2	Test also if drop down menu does not open or navigation within it is not possible	Approve Distribution creation should be successful, status change appear in Status bar and message appear in message panel					
9	**SI Text**	1	Enter SI Text for the load, different containers and also Lock positions	SI Text should be in Supp. Info Panel					
		2	Try to add there all the information that will be needed for Ramp or Freight or other users	All the needed information should be fitted for SI field					
10	**Several rows of commodities added**	1	Add one commodity	Commodity should be possible to add.					
		2	Add other commodity to the same compartment/container	Other commodity should be possible to add.					
		3	Add…	Other commodity should be possible to add.					
		4	Add commodities as long it is possible in Modify Load panel and also View panel in Load Distribution screen, Ramp module and Freight Module.	Commodities should be possible to add as much as possible or designed.					

11	**Background color of the Finnair aircraft**	1	Ensure that in every aircraft type background color is white	Background screen should be white					
		2	Check also other colours in different panels, buttons and active fields. Are colours on those correct?	Compare color to the specs.					
12	**Mandatory and non-mandatory fields**	1	Unfell or leave details incomplete (or try to fill long detail information) in some mandatory fields and test is it possible to change screen or exit panel (Pantry, Crew, Move Load, Documents Type, Cabin Load panel, Dead load screen, Dead load Details panel, etc.)	Exit or changing screens should not be possible.					
		2	Fill some non-mandatory fields (Dead load screen, Dead load Details panel, etc.) and also mandatory fields and ensure that you can exit current panel or screen	Exit or changing screens should be possible.					
13	**Moving load between containers, bulk and offload**	1	Move load between different positions, add and delete is and see what kind of details can be seen on Load Distribution screen on certain container or compartment details.	Screen update should be successful and details should occur / not occur on certain position where load is added/removed					
		2	Move load between different positions, add and delete it with Modify Load Panel, Dragging and View Panel.	Screen update should be successful and details should occur / not occur on certain position where load is added/removed					
14	**Messages**	1	Message is possible to view	Message is possible to read and sign out.					

	and messaging	2	Message will change color of the flight and envelope will appear in My Flights Panel when message is readable	Color is right depending about message priority					
		3	Message will give a pop-up when it's a high priority message	Pop-up will appear when message is high priority message.					
		5	Message will be able to read from messenger	Message is possible to read and sign out.					
		6	Message mark (envelope) will disappear from My Flights panel when it is ticked	Envelope will disappear and different color will be on the background color of the flight (green or marble) when message is marked as read and notified.					
15	**Triggering flight actions**	1	Flight action will be triggered automatically	Color of the flight status will change when triggered and the color depends is it accomplish successfully.					
		2	Flight action will be triggered manually	Color of the flight status will change when triggered and the color depends is it accomplish successfully.					
		3	Response time of the triggering	Flight action should effect for the flight immediately when it is triggered automatically or manually					
16	**Proper TAB functioning**	1	Press in different screens TAB button from the keys	Mouse pointer should change place in certain order.					

17	**Radio buttons and check boxes**	1	Radio button can be chosen only once above one topic	One radio button is only possible to choose					
		2	In certain topic check boxes can be choose with the different combination	Different combinations of the check boxes can ticked					
18	**Text fields**	1	Enter in text field text, try to find maximum value for the text	Maximum amount of the text can be inserted.					
		2	Enter in text field text or characters that should not be possible to enter this text field.	Special characters could not be entered in text fields where restriction is defined					
19	**Screen validation test cases**	1	Colours are as specified						
		2	Screens are resizable						
		3	Buttons and menus are displayed properly						
		4	Pop-ups are entire						
		5	Text is readable and font is big enough						
20	**Help screen is displayed**	1	TOC		Rope Sidebras	27.6.2007	Pass		Help menu seems to work without problems
		2	Index		Rope Sidebras	27.6.2007	Pass		
		3	Search		Rope Sidebras	27.6.2007	Pass		
		4	Previous Page button		Rope Sidebras	27.6.2007	Pass		
		5	Next Page button		Rope Sidebras	27.6.2007	Pass		
		6	Expand main text field		Rope Sidebras	27.6.2007	Pass		
		7	Re-expand main text field		Rope Sidebras	27.6.2007	Pass		

21	**Simulation Mode**	1	Run tests that will show simulation mode in different screens	Background color should change and some buttons change to disabled					
		2	If simulation mode in on, make sure that updates is not possible to make	Some of the functionalities should not work, e.g. buttons and adding/changing figures					
		3	When leaving from screen to other screen, simulation mode should stay on if selected	Simulation mode should stay on, even changing screen					
22	**Pop-up windows/ Panel testing with some panels**	1	Pop-up windows to be tested: contacts, starting weight, pantry, crew, swa, ballast fuel, dead load details, capacity calculation, cabin load, manual passenger entry, load sheet rules, publish message						
		2	Panels to be tested: defined later if needed						
		3	See if buttons, panels, fields, menus, drop-down menus, rows are working properly						

Altéa FM UAT project

Error ID	Error Description	Severity level	Date found	By whom	Date fixed	By whom	Remarks

Severity Level Definitions

When reporting defects, the following severity levels are used:

Severity Level	Description	Example
1	System Failure. No further processing is possible.	Critical to application availability, results, functionality, performance or usability.
2	Unable to proceed with selected function or dependants.	Application sub-system available, key component unavailable or functionally incorrect and workaround is not available.
3	Restricted function capability, however, processing can continue.	Non-critical component unavailable or functionally incorrect; incorrect calculation results in functionally critical key fields/dates and workaround is available.
4	Minor cosmetic change.	Usability errors; screen or report errors that do not materially affect quality and correctness of function, intended use or results.

LIITE 6: VIIKKORAPORTOINNIN MALLI ALTÉA-PROJEKTISSA

Summary test statistics for AY- UAT of Altéa DCS FM: Current week summary stats

Test Phase	2
Report for week ending:	9.11.2007
Baseline or revised test start date	3.9.2007
Baseline or Revised acceptance test end date	12.10.2007

Copy the Total row to "Weekly progress"

Area tested	Target test completion date	Number of test cases planned	Number of test cases written	% written	Number of test cases tried	% tried	Number of test cases passed	% passed	Number of PTRs opened during the week	Number of PTRs open at end of week	Blocking PTRS (Priority)	Notes
Total	**12-Oct-07**	**730**	**707**	**97 %**	**673**	**92 %**	**652**	**89 %**	**0**	**28**		
Impacted applications		97	74	76 %	67	69 %	65	67 %		2		
Basic functionality		417	417	100 %	417	100 %	410	98 %		19		
Functional gaps		113	113	100 %	113	100 %	101	89 %		7		
End to end		103	103	100 %	76	74 %	76	74 %				

Detail report of tests: Finnair
Impacted Applications

Area tested	Target test completion date	Number of test cases planned	Number of test cases written	% written	Number of test cases tried	% tried	Number of test cases passed	% passed	Number of PTRs opened during the week	Number of PTRs open at end of week	Blocking PTRs (Priority)	Notes
Totals		**97**	**74**	**76 %**	**67**	**69 %**	**65**	**67 %**		**2**		
eGO	**28-Sep-07**	**7**	**7**		**7**	**100 %**	**6**	**86 %**		**1**		**PTR raised 02375430 -** Final Nil message rejected in FM **Closed PTR 02362106 -** Prelim UWS message does not update figures in FM
DW	**5-Oct-07**	**10**	**10**		**3**	**30 %**	**3**	**100 %**				**Closed PTRs 02346354 - Archive Flight Not Succesful**

Detail report of tests: Basic functionality

Area tested	Target test completion date	Number of test cases planned	Number of test cases written	% written	Number of test cases tried	% tried	Number of test cases passed	% passed	Number of PTRs opened during the week	Number of PTRs open at end of week	Blocking PTRs (Priority)	Notes
Totals		**417**	**417**	**100 %**	**417**	**100 %**	**410**	**98 %**		**19**		
Ramp	**17-Sep-07**	**32**	**32**		**32**	**100 %**	**27**	**84 %**		**4**		**New PTR** **02341145 -** Destination not correct in Modify Load in Ramp Module **02348635** - Offloading from ramp agent window via modify load panel **02349095** - Cabin load error message when RA try to move load **02361851** - Enter key is not working in Modify load panel **Closed PTR** **02342097 -** DG/SL conflict not shown in ramp module **02211221** - Ramp module function not able to process without mouse **02350180** - Updating rest location field in modify load panel **02348692** - Error message for RA if load distribution not found

Detail report of tests: Functional Gaps (Change Requests)

Area tested	Target test completion date	Number of test cases planned	Number of test cases written	% written	Number of test cases tried	% tried	Number of test cases passed	% passed	Number of PTRs opened during the week	Number of PTRs open at end of week	Blocking PTRs (Priority)	Notes
Totals		113	113	100 %	113	100 %	101	89 %		7		
Loadsheet - AYD90	5-Oct-07	22	22		22	100 %	19	86 %		3		N/A - Confidential.
Special Baggage - AYD58	5-Oct-07	32	32		32	100 %	28	88 %		1		N/A - Confidential.

Detail report of tests: end-to-end

Area tested	Target test completion date	Number of test cases planned	Number of test cases written	% written	Number of test cases tried	% tried	Number of test cases passed	% passed	Number of PTRs opened during the week	Number of PTRs open at end of week	Blocking PTRs (Priority)	Notes
Totals		**103**	**103**	**100 %**	**76**	**74 %**	**76**	**74 %**				
End to end	**5-Oct-07**	**103**	**103**		**76**	**74 %**	**76**	**100 %**				Rest of the test cases will be executed in regression testing and parallel run

Weekly progress since start

Week ending	Target test completion date	Number of test cases planned	Number of test cases written	% written	Number of test cases tried	% tried	Number of test cases passed	% passed	Number of PTRs opened during the week	Number of PTRs open at end of week	Blocking PTRS	Comments
2.11.2007	12.10.2007	730	707	97 %	673	92 %	652	89 %	0	28		
26.10.2007	12.10.2007	730	707	97 %	673	92 %	652	89 %	0	28		
19.10.2007	12.10.2007	730	707	97 %	673	92 %	652	89 %	0	28		
12.10.2007	12.10.2007	730	707	97 %	673	92 %	652	89 %	0	28		
5.10.2007	12.10.2007	756	691	91 %	645	85 %	610	81 %	0	27		
28.9.2007	12.10.2007	801	689	86 %	466	58 %	409	51 %	0	27		
21.9.2007	12.10.2007	660	590	89 %	250	38 %	230	35 %	0	22		
14.9.2007	12.10.2007	660	590	89 %	128	19 %	116	18 %	0	16		
7.9.2007	25.10.2007	660	590	89 %	0	0 %	0	0 %	0	14		

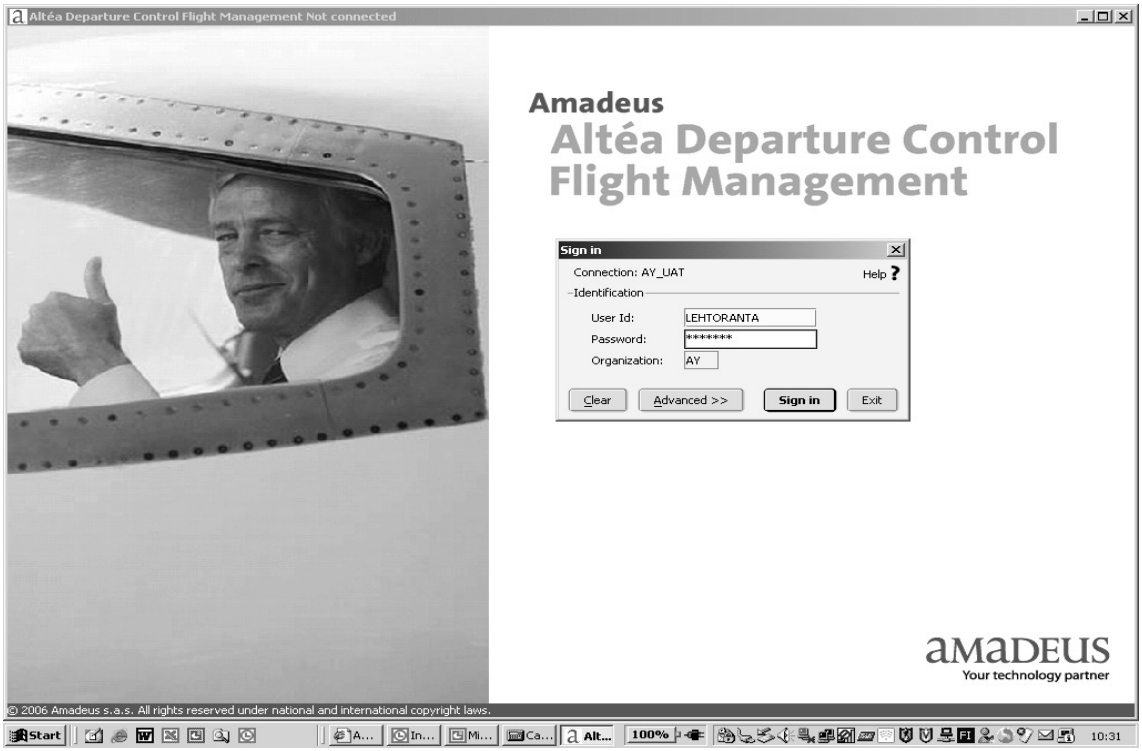
LIITE 7: ALTÉA-PROJEKTISSA KÄYTETYN TESTAUSSTRATEGIAN SISÄLLYSLUETTELO

Table of Contents

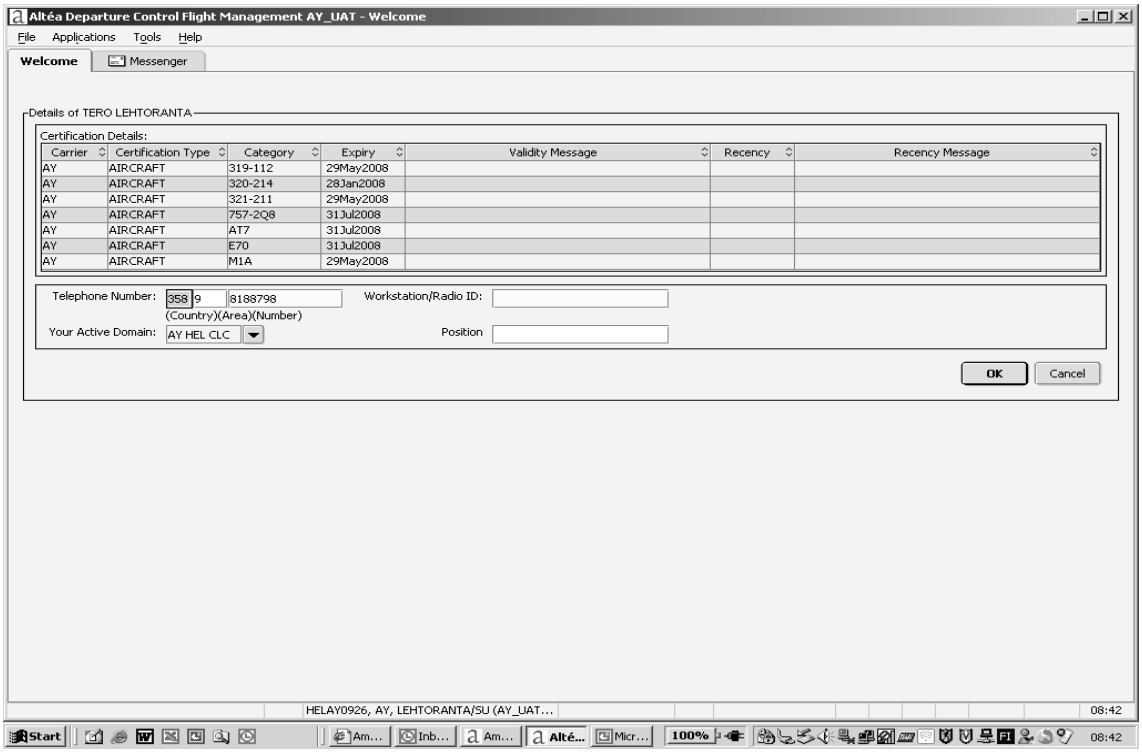
LIITE 8: ALTÉA-PROJEKTISSA KÄYTETTY TESTAUSSUUNNITELMAN SISÄLLYSLUETTELO

Table of Contents

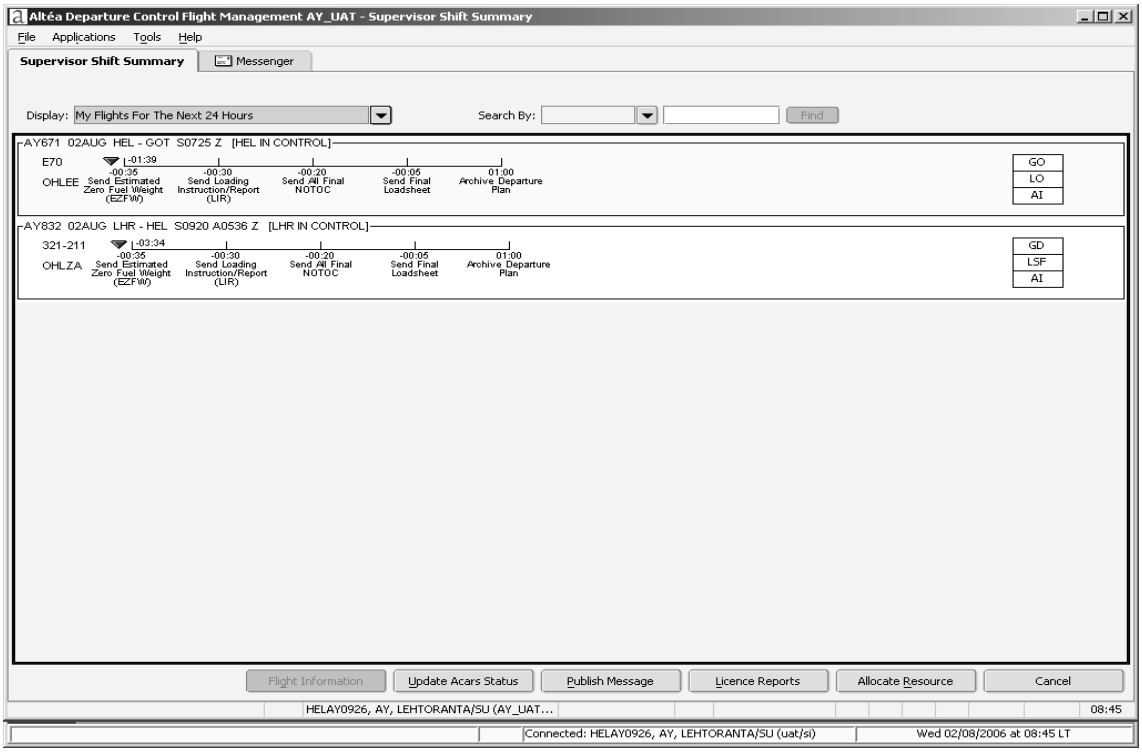
LIITE 9: NÄYTTÖJÄ ALTÉA FM -OHJELMISTOSTA



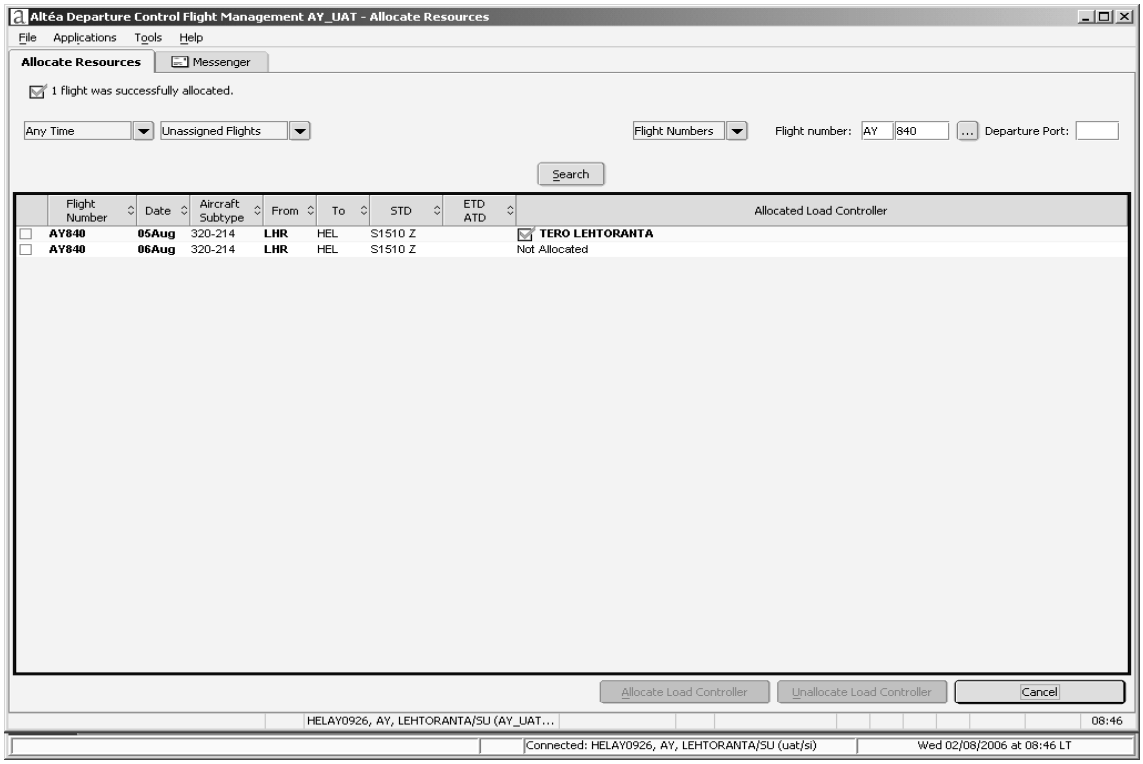
Kuva 1: *Kirjautumisikkuna.*



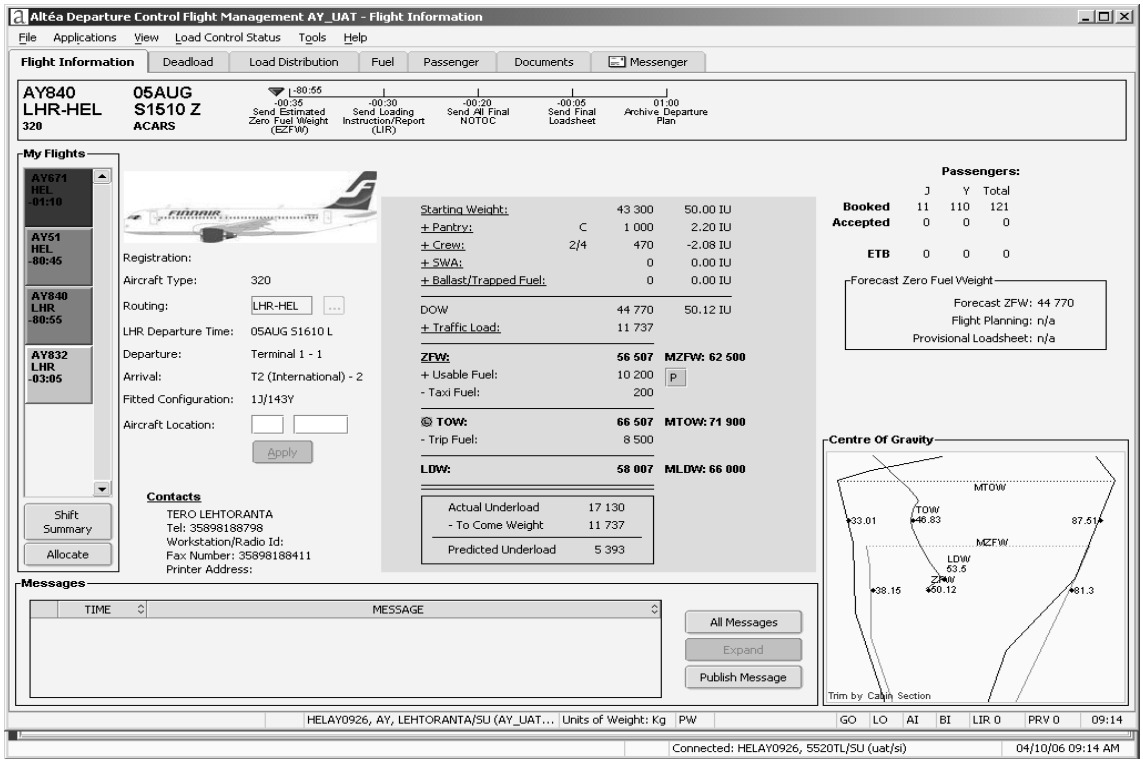
Kuva 2: *Tervetuloa-näyttö kirjautumisen jälkeen.*



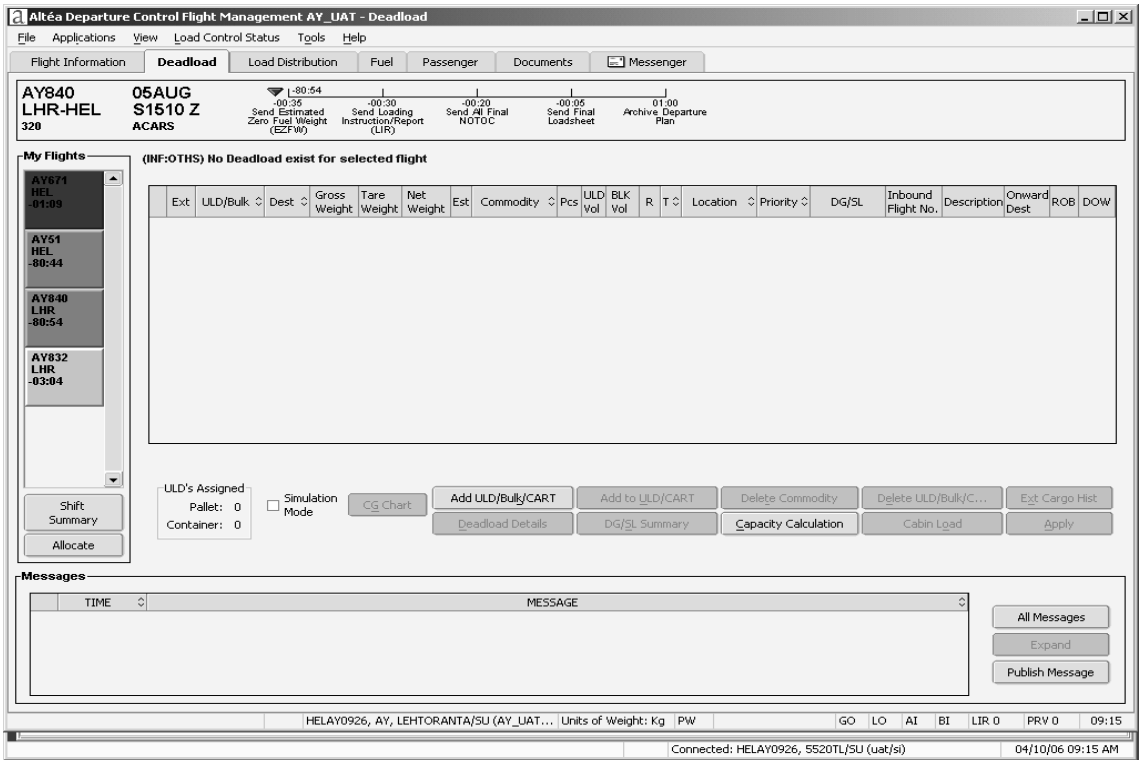
Kuva 3: *Lentojen valvonta.*



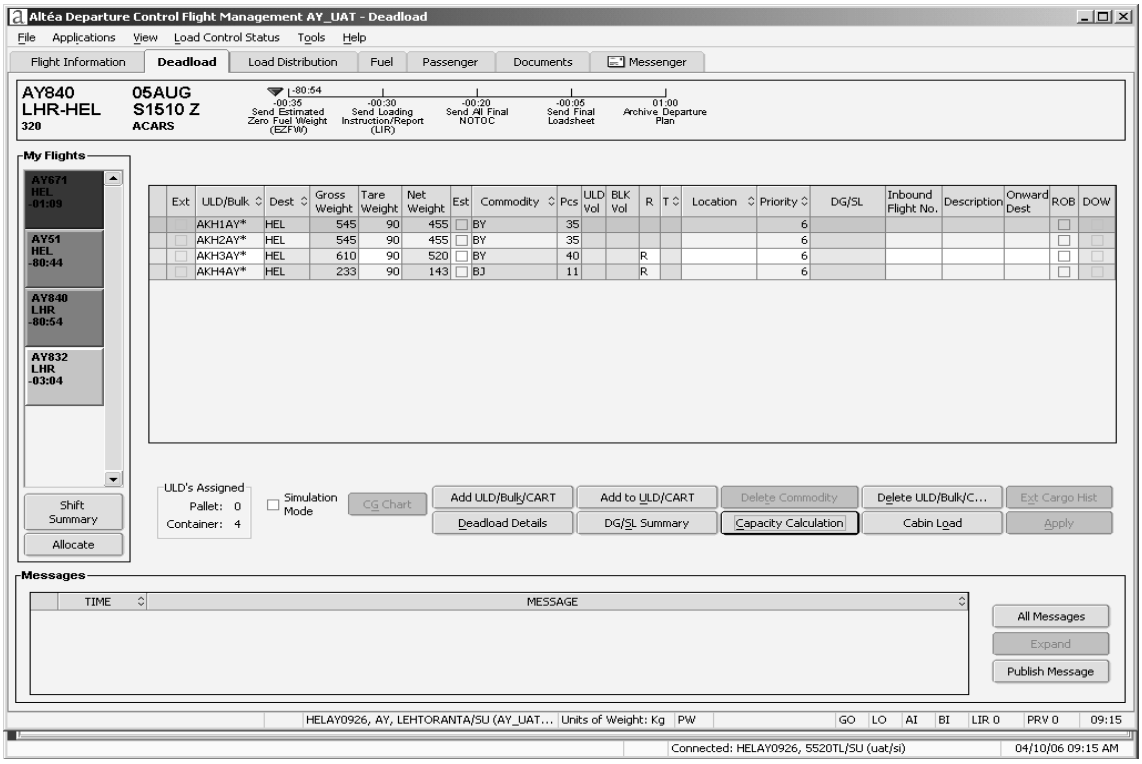
Kuva 4: *Lentojen allokointi.*



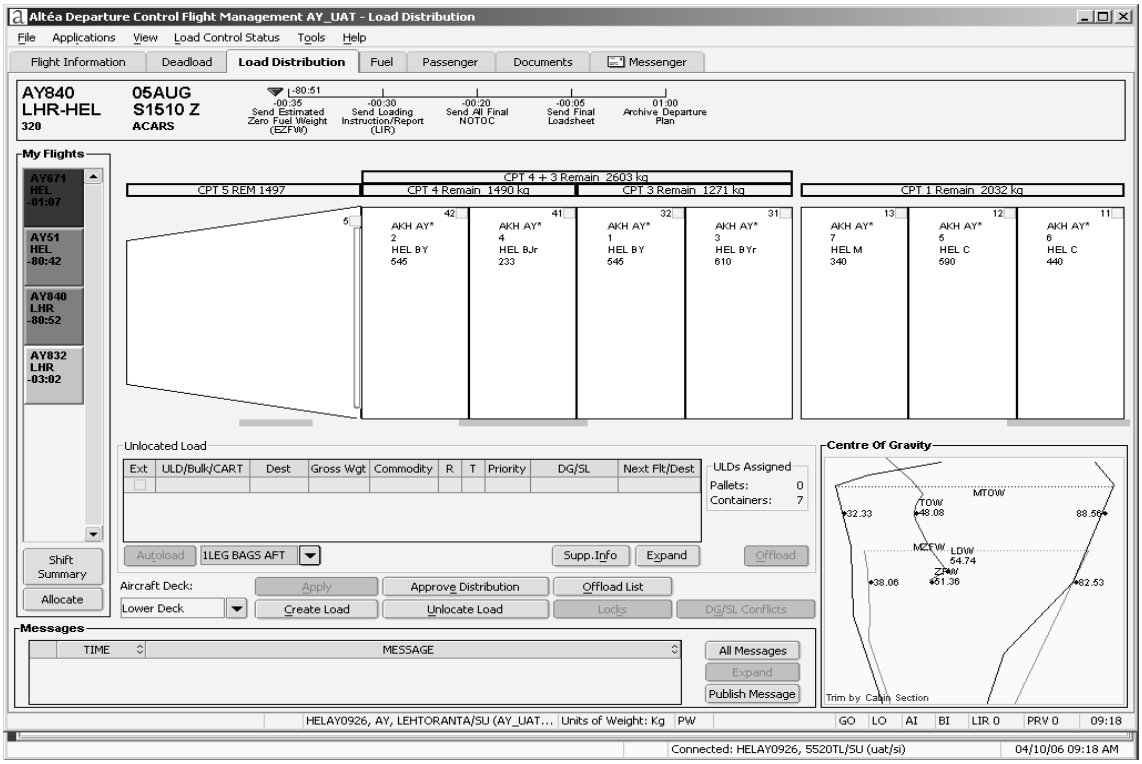
Kuva 5: *Lennon tietoikkuna, josta pääsee käsiksi kaikkiin lennon toiminnallisuuksiin ja Altéa FM -moduuleihin.*



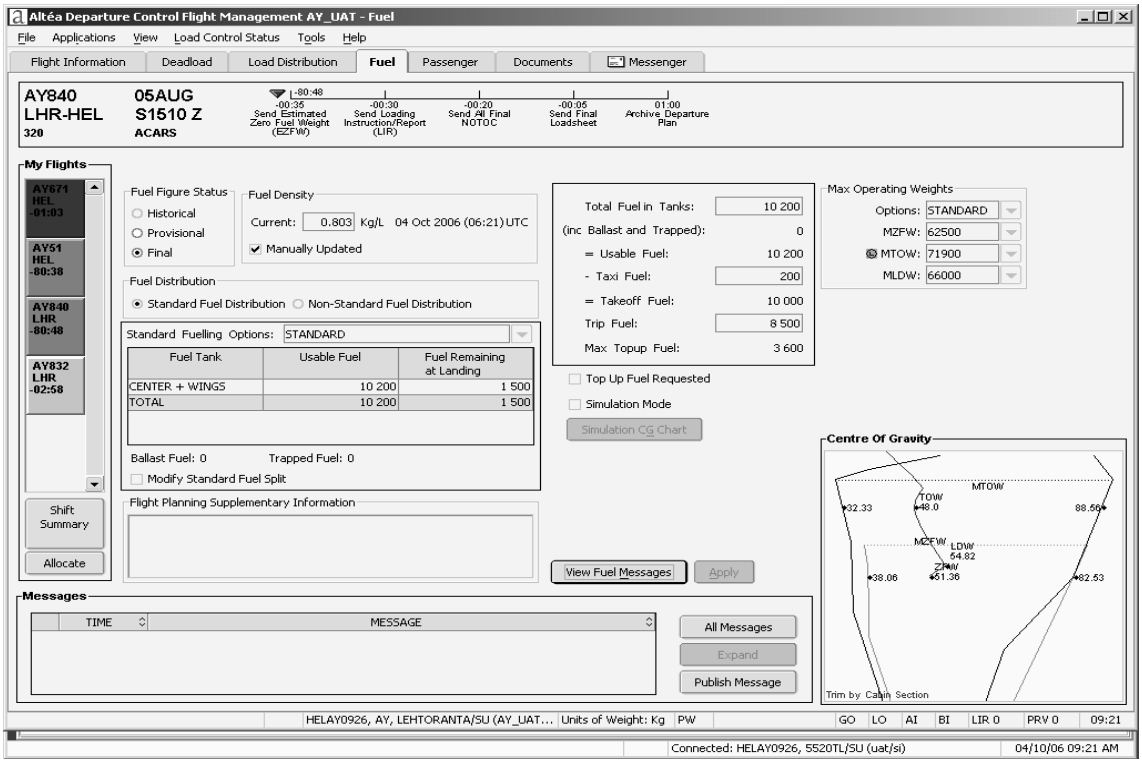
Kuva 6: *Kuorman näyttö (Deadload).*



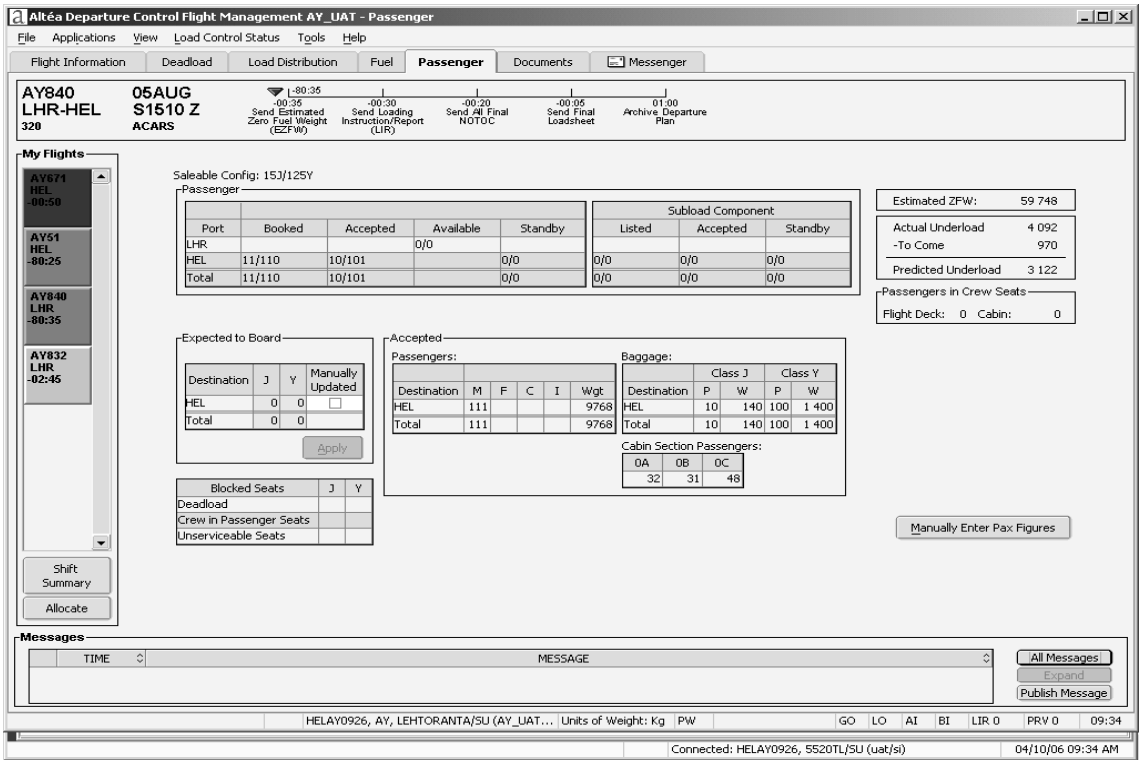
Kuva 7: *Kuorman näyttö päivitettynä matkatavaratiedoilla.*



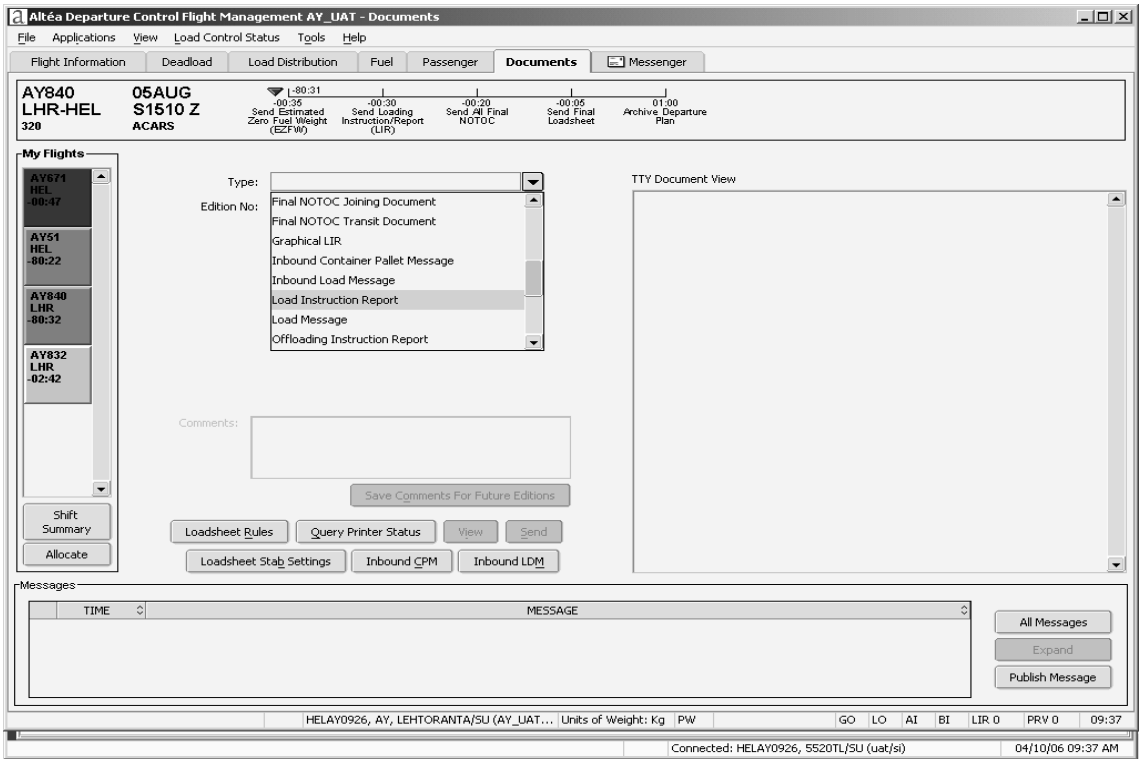
Kuva 8: *Kuormausohje graafisena esityksenä (Load Distribution).*



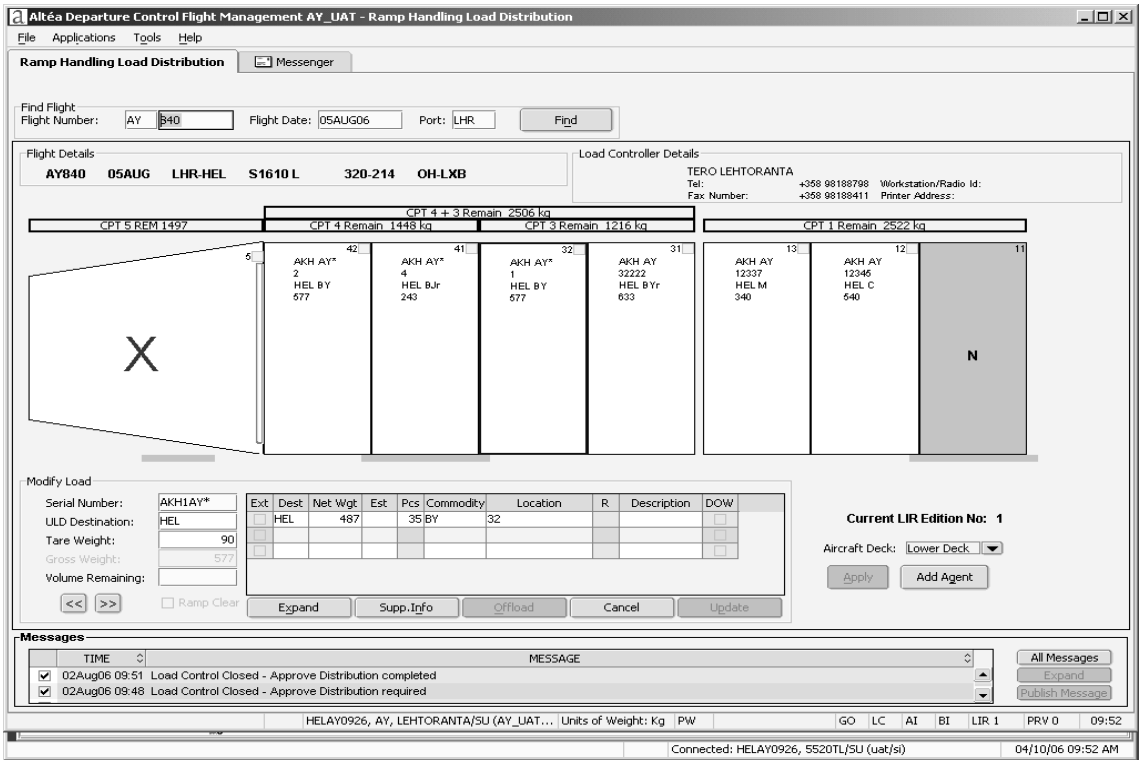
Kuva 9: *Polttoaine-näyttö ja polttoaineen tiedot.*



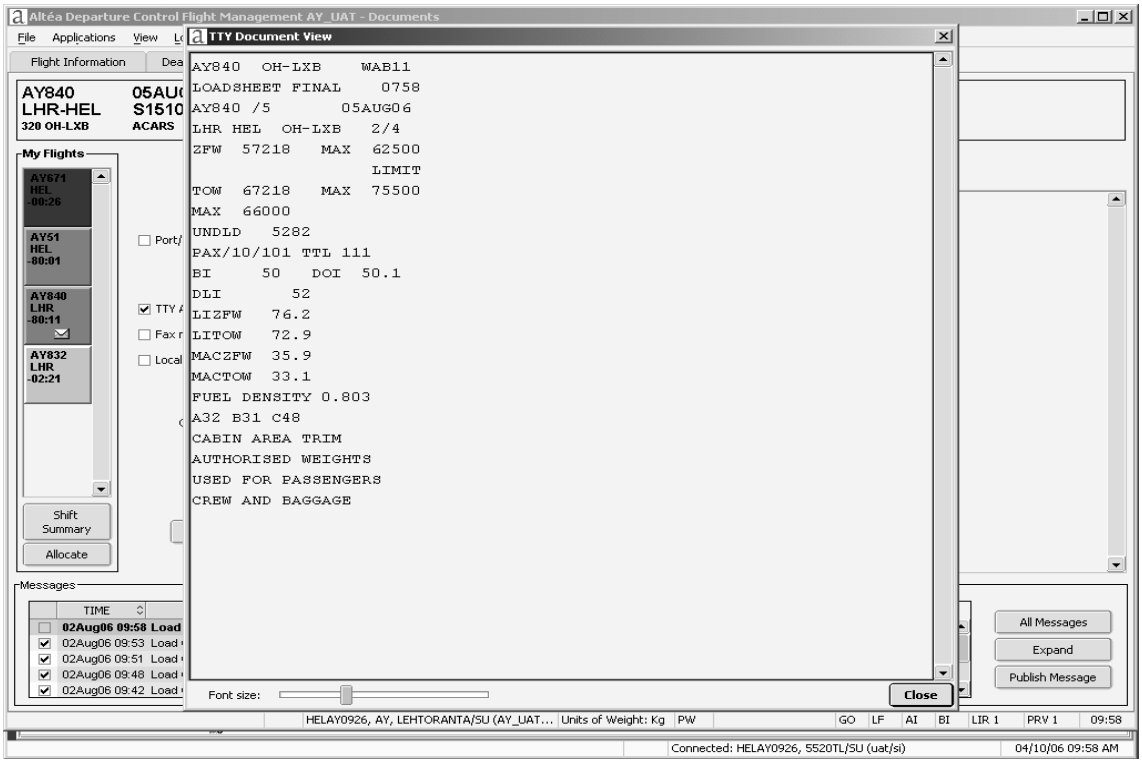
Kuva 10: *Matkustajatiedot.*



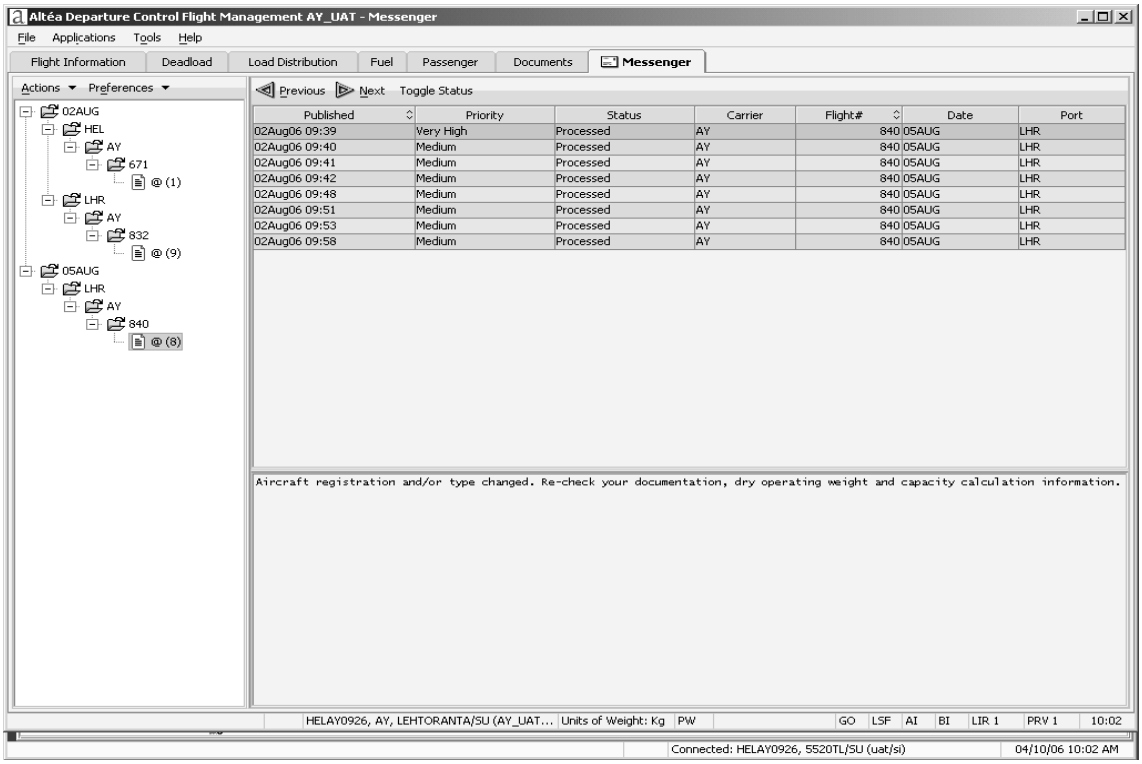
Kuva 11: *Dokumenttinäyttö ja niiden lähetystoiminnallisuus.*



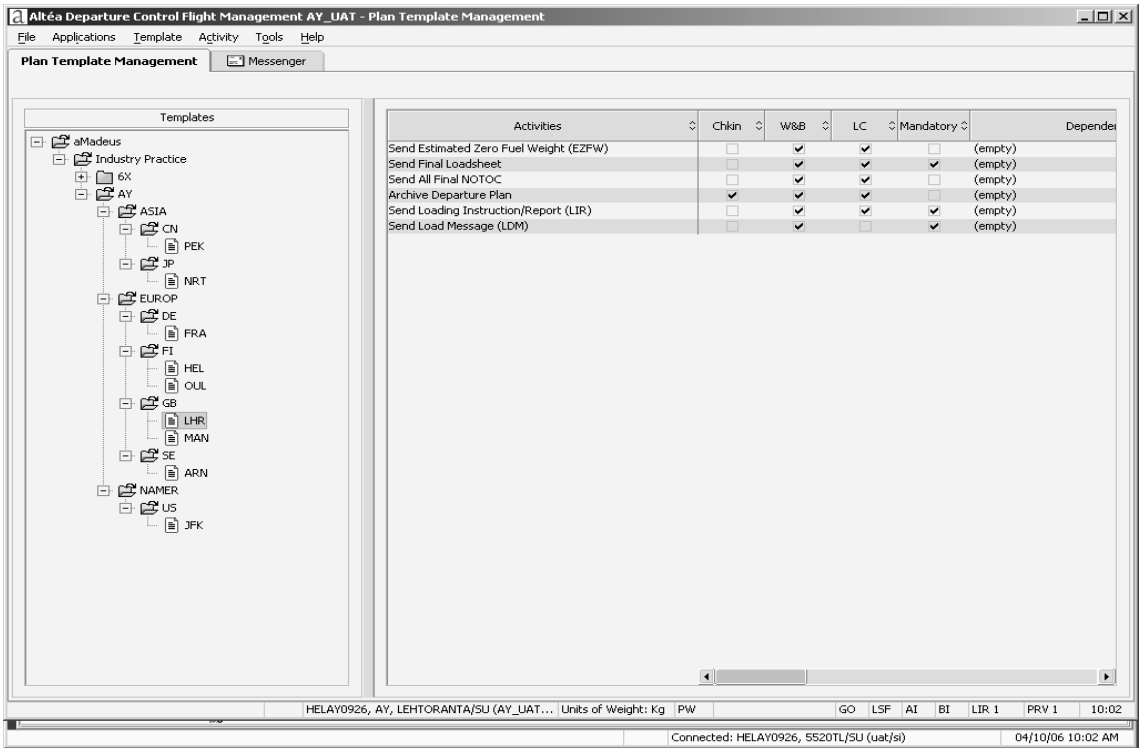
Kuva 12: *Kuormauksen kuittaus koneen lastauksen jälkeen.*



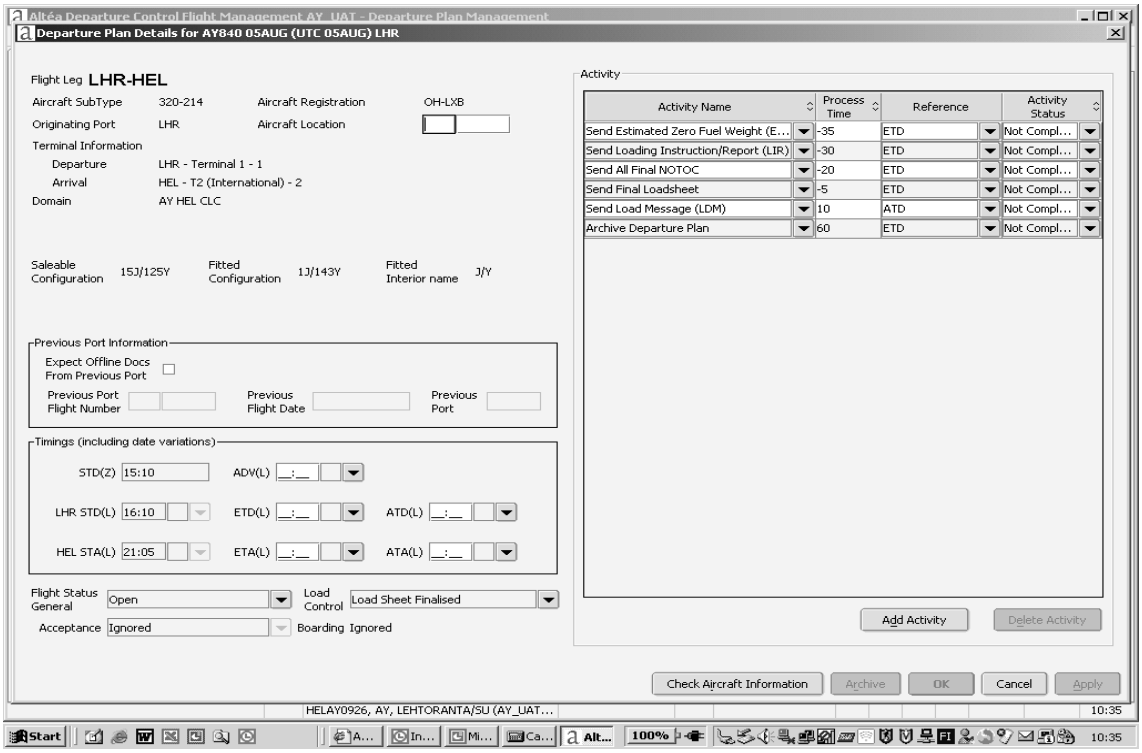
Kuva 13: *Tasapainolaskelmien pohjalta lasketut luvut, eli ohjeistus koneen kapteenille lennon lähtöä varten (Final Loadsheet).*



Kuva 14: *Viesti-ikkuna.*



Kuva 15: *Lentojen lähtöä varten suunniteltu vaiheistus lennon valmistelua varten.*



Kuva 16: *Lentojen lähtöä varten suunnittelutoiminnallisuus eri vaiheiden automatisoinnille tai tarvittaville varoituksille.*